Shri Krishna College of Management & Technology

2nd National Conference

On

Advance Research in Computer Science & Engineering

(ARCSE- 2019)

4th & 5th September 2019

Organized by

Department of Computer Science & Engineering

Venue

NH-58 Shahbuddinpur Chowk, Roorkee Road, Muzaffarnagar, (U.P.) 251001

CHIEF PATRONS

Mr. Harish Gautam

ADVISORY COMMITTEE

Er. Ankit Kumar (B.Tech, M.Tech)
Er. Ravindra Kumar Pal (B.Tech)
Er. Ashwani Kumar Sharma (M.Tech)
Er. Prahlad Tomar (M.Tech)
Mr. Shrikant Tyagi (MBA)
Mr. Praveen Pal (Physics & M.Tech)
Ms. Shivani Sharma (MA & B.Ed)
Ms. Rashmi Tyagi (BE)
Mr. Vinay Kumar Panchal (Diploma, B.Tech)

TECHNICAL COMMITTEE

Mr. Gaurav Kumar Uppadhyay
Er. Harsh
Er. Shivam Sharma
Er. Arun Pal
Er. Annu Meena
Er. Devesh Patel
Er. Savindra Singh
Mr. Rehan
Mr. Ravi Prakash

ORGANIZING COMMITTEE

Convener	Co-Convener
Ms. Meenakshi Pal	Er. Dhananjay Singh

Organizing Secretary

Er. Krishn Pal Singh

A Software Oriented Approach to Solve Some Transportation Problem

Manoj Kumar
Research Scholar, Department of Mathematics
Bhagwant University
Ajmer, Rajasthan, India
manojsharma8383@gmail.com

Dr. Ajay Kumar Gupta
Department of Mathematics
Bhagwant University
Ajmer, Rajasthan, India

Abstract— **The MATLAB coding method is better than analytical method for solving Transportation Problem (TP). The model gives a very good result in Transportation problem (TP).**

In this paper we are presenting a software approach to solve some transportation problem using computational performance and techniques. In this paper we are solve some transportation problem with the help of C++ and Matlab.

The transportation model starts with the event of a possible answer, that is then consecutive tested and improved till a best answer is obtained.

Keywords—TP; Matlab; C++; Method; Analytic

I. Introduction

Now days MATLAB is very mostly used mathematics such as MATLAB with Differential Equation, Numerical methods, Operation Research, Fuzzy Logic etc. In this paper MATLAB coding is used to solve Transportation Problem. This gives optimal solution within fraction of seconds.

The Transportation drawback involves finding the lowest-cost arrange for distributing stocks of products or provides from multiple origins to multiple destinations that demand the products. The transportation model may be wont to determine a way to portion the provides on the market from the assorted factories to the warehouses that stock or demand those product, in such some way that total shipping price is reduced. Usually, analysis of the matter can manufacture a shipping arrange that pertains to a precise amount of your time (day, week), though once the arrange is established, it will generally not amendment unless one or additional of the parameters of the matter (supply, demand, unit shipping cost) changes.

It is developed and published by Harold W.Kuhn(1995), who gave the name "Hungarian Method" because the algorithm was largely based on the earlier works of two Hungarian Mathematicians: Denes Konig and Jeno Egervary. Moreover MATLAB is a software package. MATLAB stands for MATrix LABoratory, it deals with matrix (array).

The transportation model starts with the event of a possible answer, that is then consecutive tested and improved till a best answer is obtained. The outline of the technique on the subsequent pages focuses on every of the most important steps within the method during this order:

II. Transportation Problem (Matlab Program)

```
function [x,cost]=transport(s,d,c);
% [x,cost]=transport(s,d,c)
% Input:
% s = supplies (m*1)
% d = demands (n*1)
% c = costs (m*n)
% Output
% x = optimal solution (m*n)
% cost = minimal transport cost

function [x,b]=northwest(s,d)
% [x,b]=northwest(s,d)
% x: shipments using nwrule
(m*n)
% b: 1 for each basic variables 0 for nonbasic (m*n)
% s: supplies (m*1)
% d: demands (n*1)
if (sum(s)~=sum(d)),
disp('ERROR: The total supply is not equal to the total
demand.');
return;
end
m=length(s);
n=length(d);
i=1;
j=1;
x=zeros(m,n);
b=zeros(m,n);
while ((i<=m) & (j<=n))
if s(i)<d(j)
x(i,j)=s(i);
b(i,j)=1;
d(j)=d(j)s(
i);
i=i+1;
else
x(i,j)=d(j);
b(i,j)=1;
s(i)=s(i)d(j);
j=j+1;
end
end
```

```
example511; [x cost] = transport(s,d,c)
 x =


     100       0      20       0
       0      60      60      20
       0       0       0     100
 cost =
                1900
```

```
example513; [x cost] = transport(s,d,c)
x =
      0       0       0      30      70
     20      60      80       0       0
     30       0       0      70       0
cost =
            1730
```

III. Pseudo code for the Transportation problem is written in Mat Lab

Items	Destination					Demand
	A	B	C	D	E	
F	7000 6	2000	6000	11000	90000	6
G	4000 94	5000 6	12000	8000	22000	100
H	9000	14000 194	21000 6	13000	19000	200
I	1000 0	18000	17000 44	25000 6	27000	50
J	3000	20000	26000	20000 94	28000 6	100
Supply	100	200	50	100	6	456

The total transportation cost is given as
7000*6+4000*94+5000*6+14000*194+21000*6+17000*44+25000*6+20000*94+28000*6= 62,36,000

```
clc;
clear all;
close all;
x=input('enter the transportation matrix');
[m n]=size(x);
x1=zeros(m,n);
sumc=0;
sumr=0;
for i=1:m-1
sumc=sumc+x(i,n);
end
for j=1:n-1
sumr=sumr+x(m,j);
end
if(sumc == sumr)
for i=1:m
for j=1:n
x11=min(x(i,n),x(m,j));
x1(i,j)=x11;
x(i,n)=x(i,n)-x11;
x(m,j)=x(m,j)-x11;
end
end
else disp('unbalanced transportation');
end
xre=0;
for i=1:m-1
for j=1:n-1
xre=xre+(x(i,j).*x1(i,j));
end
end
disp(['the transportation cost is ',num2str(xre)]);
```

IV. Conclusion

The system developed in this study was desktop-based which is limited to usage by only users who have the software installed on their computers. It will however be of more advantage if the system is made webbasedas this will enable different users access the system from different locations and with different devices.
It is a novel approach to solve the TP. We use MATLAB software to solve the transportation problems. It is very simple code write in Mat lab and this is a very simple programming languages. Transportation problem solve very easy through Matlab programming and results are very accurate.

References

[1]. Abdul , S. S., Muhammad , J., & Gurudeo , T. A. (2015). Modified Vogel's Approximation Method For Solving. *Mathematical Theory and Modeling, 5*, 32-42.

[2]. Alexander, R., & Schrijver, U. (1998). *Theory of Linear and Integer Programming.* John Wiley & Sons.

[3]. Crainic, T. G. (1998). Network Design in Freight Transportation. *IFIP NG 7.6-IIASA Workshop on Advances in Modelling: Paradigm Methods and Application.* Austria. Retrieved from http://www.sciencedirect.com/science/article/pii/S0377221799002337

[4]. Dalgobind, M. (2015). *The MODI and VAM Methods of Solving Transportation Problems.* Retrieved from http://www.academia.edu/23726290/MODI_and_VAM_Methods

[5]. Elsie, N. (2007). *Introduction to Operations Research.* Aba: She Nox Publishers.

[6]. Eze, J. I., Obiegbu, M. E., & Jude-Eze, E. N. (2005). *Statistics And Quantitative Methods for Construction And Business Managers.* Lagos: The Nigerian Institute of Building.

[7]. Stormy Attaway, MATLAB: A Practical Introduction to Programming and Problem Solving, Elsevier, Inc 2009.

[8]. Keng - Cheng Ang, Introducing the boundary element method with MATLAB, Int. J.Math. Edu. Sci. and Technol.

[9]. P. Dunn and C. Harman, Calculus demonstrations using Matlab, Int. J. Math. Edu. Sci. and Technol. 33 (2002)

[10]. L. Colgan,Matlab in first - year engineering mathematics, Int. J.Math. Edu. Sci. and Technol. 31 (2000) 15-25.

[11]. E. Tonkes, B. Loch, and A. Stac, An innovative learning model for computation in first year mathematics, Int. J.

With the Inspection Policy Availability Analysis of Cables Used as a 2- Unit Cold Standby System in Metro Railways

Ranu Pandey[a] , B. Parashar[b], A.K. Gupta[c]

[a, c]*Bhagwant University, Ajmer, 305004, India.*

[b]*JSS Academy of Technical Education, Noida, 201301, India.*

Abstract: The present study of the metro network system necessitates carrying out availability of this metro railways system through power cables. These cables are laid in loops for feeding supply to substations from each station. Initially one cable is in operation and the other cable of same voltage capacity is taken as cold standby. In order to find out the type of failure i.e. whether repairable or irreparable, Inspection has been carried out at each failure. The failure data have been collected from the cable manufacturing company and attempt has been made to obtain Availability Analysis (AA) of the system using semi-Markov processes and regenerative point technique. The Graphical analyses have also been made for Availability with respect to the failure rate of the present system.

Keywords: Power Cables, system effectiveness, semi-Markov processes, regenerative point technique

1. Introduction

The study of reliability results in taking recourse for effective utilization and timely maintenance of different industrial engineering systems. Large amount of work on reliability and profit analysis has been carried out on different types of standby redundant systems by various researchers including [1-4]. B. Parashar and G. Taneja [5] analyzed a PLC system based upon real data collected from industry. Z. Zhang et al. [6] and G. Taneja [8] discussed aspects of maintenance and variation in demand of different systems. B. Parashar & A. Naithani [7&10] studied a system of ID fans in a thermal plant wherein different conditions making the system work at full/reduced capacity discussed. To maintain the system to work continuously and efficiently, it becomes significantly of high importance that the failure whether due to cable manufacturing defects or some system malfunctioning, gets reduced to lowest level [9]. In the metros network, various power cables with rated capacities of 220, 132, 66, 33, 25 KV are used in supplying power for the functioning of the system. The failure data of 12-13 years have been collected for the power cable with the capacity of 33KV. The aim is to analyse the availability of the system by considering a two unit cold standby identical parallel power cable of 33KV capacity. The steady state Availability Analysis of system effectiveness is analyzed by making use of semi-Markov processes and regenerative point technique.

2. Symbols and Notations

O	Operative state of the power cable
O_{cs}	Cold standby unit of the power cable
λ	Constant failure rate of the operative power cable
p	Probability of failure (repairable) of the unit
q	Probability of replacement (irreparable) of the unit
r	Probability of the unit found Ok
F_{ui}	Unit is under inspection in case of failure
F_r	Unit is under repair in case of major failure
F_{rp}	Unit is under replacement
F_{Ui}	Continuation of inspection of failure from its previous state.
F_R	Continuation of repair of failure from its previous state.

F_{RP}	Continuation of replacement from its previous state.
F_{wi}	Failed unit waiting for inspection
γ	Constant rate of inspection
α	Constant rate of repairable failure
β	Constant rate of replacement failure
$g(t), G(t)$	p.d.f. and c.d.f. of repair time of unit having major failure
$h(t), H(t)$	p.d.f. and c.d.f. of replacement time of unit having major failure
$h_1(t), H_1(t)$	p.d.f. and c.d.f. of inspection time of unit having failure
$w(t), W(t)$	p.d.f. and c.d.f. of waiting time of a failed unit
$q_{ij}(t), Q_{ij}(t)$	p.d.f. and c.d.f. of first passage time from a regenerative state i to j or to a failed state j without visiting any other regenerative state in $(0,t]$
p_{ij}	Transition probability from regenerative state i to regenerative state j
$M_i(t)$	Probability that system up initially in regenerative state i is up at time t without passing through any other regenerative state
m_{ij}	Contribution to mean sojourn time in regenerative state i before transiting to regenerative state j without visiting to any other state
$\mu_i(t)$	Mean sojourn time in regenerative state before transiting to any other state
©, $\otimes$	Symbols for Laplace and Laplace Stieltje's convolution
*, **	Symbols for Laplace and Laplace Stieltje's transforms
A_0	Steady state availability of the system

3. Data Summary

The following values have been obtained from the collected data:

- Estimated value of failure rate (λ) =.000015 per hour
- Estimated value of repair rate (α) =.067per hour
- Estimated value of replacement rate (β) =.002 per hour
- Estimated value of inspection rate (γ) =1 per hour
- Probability of repairable failure (p) =.69
- Probability of replaceable failure (q) =.16
- Probability of unit found ok (r) =.15

4. Model Description and assumptions

The following assumptions have been used in probabilistic model:

1) Initially the system is operative at full capacity with two power cables, one is operative and other one is used as a cold standby of same capacity.
2) The two power cables are identical and constitute a parallel system.
3) If one unit is failed, standby unit becomes operative after some time.
4) As the unit fails, it is undertaken for inspection.
5) The failure rate of hot standby unit is same as that of operative unit.
6) Failure, repair and inspection times are assumed to follow an exponential and general time distribution respectively.

7) The common repairman is available for the defaulted cable.
8) The repaired unit works as good as new one. When both units are failed then the system is in failed state.
9) All the random variables are independent.

5. Transition Diagram

A state transition table showing all the possible states of the system under consideration as shown in the Fig.1. The states 0, 1, 3, 4, 5, 7 are regeneration points and thus these are called regenerative states. The states 0, 1, 5, 7 are up states where as 2, 3, 4, 6, 8 are failed states.

States	0	1	2	3	4	5	6	7	8
0	-	2λ	-	-	-	-	-	-	-
1	$rh_1($	-	λ	-	-	$ph_1($	-	$qh_1($	-
2	-	$rh_1($	-	$ph_1($	$qh_1($	-	-	-	-
3	-	$g(t$	-	-	-	-	-	-	-
4	-	$h(t)$	-	-	-	-	-	-	-
5	$g(t$	-	-	-	-	-	λ	-	-
6	-	$g(t$	-	-	-	-	-	-	-
7	$h(t)$	-	-	-	-	-	-	-	λ
8	-	$h(t)$	-	-	-	-	-	-	-

Fig.1: State Transition Table

6. Transition Probabilities and Mean Sojourn Times

The steady-state transition probabilities are

$$dQ_{01}(t) = 2\lambda e^{-2\lambda t}dt,$$

$$dQ_{12}(t) = \lambda e^{-\lambda t}\bar{H}_1\,dt,$$

$$dQ_{10}(t) = re^{-\lambda t}h_1(t)dt,$$

$$dQ_{15}(t) = pe^{-\lambda t}h_1(t)dt,$$

$$dQ_{17}(t) = qe^{-\lambda t}h_1(t)dt,$$

$$dQ_{11}^2(t) = r(1-e^{-\lambda t})\mathrm{h}_1(t)dt,$$

$$dQ_{14}^2(t) = q(1-e^{-\lambda t})h_1(t)dt,$$

$$dQ_{13}^2(t) = p(1-e^{-\lambda t})h_1(t)dt,$$

$$dQ_{50}(t) = e^{-\lambda t}\mathrm{g}(t)dt,$$

$$dQ_{56}(t) = \lambda e^{-\lambda t}\bar{G}(t)dt,$$

$$dQ_{51}^6(t) = (\lambda e^{-\lambda t}\copyright 1)\mathrm{g}(t)dt,$$

$$dQ_{71}^8(t) = (\lambda e^{-\lambda t}\copyright 1)h(t)dt,$$

$$dQ_{70}(t) = e^{-\lambda t}h(t)dt,$$

$$dQ_{78}(t) = \lambda e^{-\lambda t}\bar{H}(t)dt,$$

$$dQ_{31}(t) = g(t)dt,$$

$$dQ_{41}(t) = h(t)dt$$

(1)

The non-zero elements $p_{ij} = \lim_{s\to 0} q_{ij}^*(s)$ can be obtained as

$$p_{01} = 1,$$

$$p_{31} = 1,$$

$$p_{41} = 1,$$

$$p_{10} = rh_1^*(\lambda),$$

$$p_{15} = ph_1^*(\lambda),$$

$$p_{17} = qh_1^*(\lambda),$$

$$p_{11}^{2} = r[1-h_1^*(\lambda)],$$

$$p_{12} = [1-h_1^*(\lambda)],$$

$$p_{13}^{2} = p[1-h_1^*(\lambda)],$$

$$p_{14}^{2} = q[1-h_1^*(\lambda)],$$

$$p_{51}^{6} = [1-g^*(\lambda)],$$

$$p_{50} = g^*(\lambda),$$

$$p_{56} = [1-g^*(\lambda)],$$

$$p_{70} = h^*(\lambda),$$

$$p_{78} = [1-h^*(\lambda)],$$

$$p_{71}^{8} = [1-h^*(\lambda)],$$

(2)

From the above steady-state transition probabilities, it can be verified that

$$p_{01} = p_{31} = p_{41} = 1,$$

$$p_{10} + p_{12} + p_{15} + p_{17} = p_{10} + p_{15} + p_{17} + p_{13}^{2} + p_{14}^{2} + p_{11}^{2} = 1,$$

$$p_{50} + p_{56} = p_{50} + p_{51}^{6} = p_{70} + p_{78} = p_{70} + p_{71}^{8} = 1.$$

(3)

The mean sojourn time (μ_i) in the regenerative state 'i' is

$$\mu_0 = \frac{1}{2\lambda},$$

$$\mu_1 = \frac{1-h_1^*(\lambda)}{\lambda},$$

$$\mu_3 = \int_0^\infty t g(t)\,dt,$$

$$\mu_4 = \int_0^\infty t h(t)\,dt,$$

$$\mu_5 = \frac{1-g^*(\lambda)}{\lambda},$$

$$\mu_7 = \frac{1-h^*(\lambda)}{\lambda},$$

$$\kappa_1 = \int_0^\infty t h_1(t)\,dt,$$

(4)

The unconditional mean time taken by the system to transit for any regenerative state j when it is counted from the epoch of entrance into state i is mathematically stated as

$$m_{ij} = \int_0^\infty t\,dQ_{ij}(t) = -q_{ij}^{*'}(0).$$

(5)

Thus,

$$m_{01} = \mu_0,$$

$$m_{31} = m_{50} + m_{51}^{6} = \mu_3,$$

$$m_{50} + m_{56} = \mu_5,$$

$$m_{10} + m_{12} + m_{15} + m_{17} = \mu_1,$$

$$m_{70} + m_{78} = \mu_7,$$

$$m_{70} + m_{71}^{8} = m_{41} = \mu_4,$$

$$m_{10} + m_{15} + m_{17} + m_{13}^{2} + m_{14}^{2} + m_{11}^{2} = \kappa_1.$$

where,

$$\kappa_1 = \int_0^\infty \bar{H}_1(t)dt = \int_0^\infty th_1(t)dt,$$

$$\mu_3 = \int_0^\infty \bar{G}(t)dt = \int_0^\infty tg(t)dt,$$

$$\mu_4 = \int_0^\infty \bar{H}(t)dt = \int_0^\infty th(t)dt.$$

7. Measure of System effectiveness

7.1. Availability Analysis

Let $A_i(t)$ be the probability that the system is working at instant time t, given that the system entered regenerative state i at $t = 0$.

Then,

$$A_0(t) = M_0(t) + q_{01}(t) © A_1(t),$$

$$A_1(t) = M_1(t) + q_{10}(t) © A_0(t) + q_{15}(t) © A_5(t) + q_{17}(t) © A_7(t) + q_{11}^2(t) © A_1(t) + q_{13}^2(t) © A_3 + q_{14}^2(t) © A_4(t),$$

$$A_3(t) = q_{31}(t) © A_1(t),$$

$$A_4(t) = q_{41}(t) © A_1(t),$$

$$A_5(t) = M_5(t) + q_{51}^6(t) © A_1(t) + q_{50}(t) © A_0(t),$$

$$A_7(t) = M_7(t) + q_{71}^8(t) © A_1(t) + q_{70}(t) © A_0(t).$$

(7)

Where,

$$M_0(t) = e^{-2\lambda t}, M_1(t) = e^{-\lambda t}\bar{H}_1(t), M_5(t) = e^{-\lambda t}\bar{G}(t), M_7(t) = e^{-\lambda t}\bar{H}(t).$$

Taking Laplace transforms of above equations and solving them for $A_0^*(s)$, we get

$$A_0^*(s) = \frac{N_1(s)}{D_1(s)},$$

The availability A_0 in steady-state of the present system is define as

$$A_0 = \lim_{s\to 0} s.\frac{N_1(s)}{D_1(s)} = \frac{N_1(0)}{D_1'(0)} = \frac{N_1}{D_1},$$

Where,

$$N_1 = \mu_0(1 - p_{11}^2 - p_{13}^2 p_{31} - p_{41}p_{14}^2 - p_{15}p_{51}^6 - p_{71}^8 p_{17}) + \mu_1 + \mu$$
$$D_1 = 1 + p_{17}\mu_4 + p_{13}^2\mu_3 + \mu_4 p_{14}^2 + p_{15}\mu_3 + \mu_0(p_{15}p_{50} + p_{17}p_{70}$$

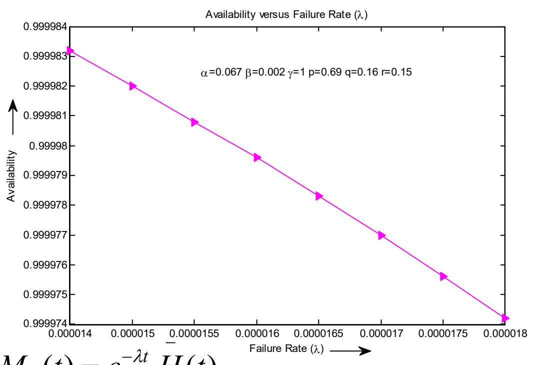
(8)

9. Particular Case

For the particular case, the inspection, repair and replacement rate are assumed to be exponentially distributed, let us take $g(t) = \alpha e^{-\alpha t}; h(t) = \beta e^{-\beta t}; h_1(t) = \gamma e^{-\gamma t}$

Using the values, as estimated in section 3, of various probabilities and repairable/replaceable rates the following measure of system effectiveness are obtained as:

Availability of the system A_0: .9998820



The Graphical Analysis

Fig. 2: Availability versus Failure rate

The following interpretations have been made from the graphs.

i. Figure 2 displays the Availability of the system decreases with increase in the failure rate λ.

References:

[1] H. Mine and H. Kawai, “Repair priority effect on avail ability of two-unit system,” *IEEE Transactions on Reliability*, vol. 28, no. 4, pp. 325–326, 1979.
[2] K. Murari and V. Goyal, “Reliability system with two types of repair facilities,” *Microelectronics Reliability*, vol. 23, no. 6, pp. 1015–1025, 1983.
[3] R. K. Tuteja, R. T. Arora, and G. Taneja, “Analysis of a two-unit system with partial failures and three types of repairs,” *Reliability Engineering & System Safety*, vol. 33, no. 2, pp. 199–214, 1991.
[4] P. Chandrasekhar, R. Natarajan, and V. S. Yadavalli, “A study on a two unit standby system with Erlangian repair time,” *Asia-Pacific Journal of Operational Research*, vol. 21, no. 3, pp. 271–277, 2004.
[5] B. Parashar and G. Taneja, “Reliability and profit evaluation of a PLC hot standby system based on a master-slave concept and two types of repair facilities,” *IEEE Transactions on Reliability*, vol. 56, no. 3, pp. 534–539, 2007.
[6] Z. Zhang, W. Gao, Y. Zhou, and Z. Zhiqiang, “Reliability modeling and maintenance optimization of the Diesel system in Locomotives,” *Maintenance and Reliability*, vol. 14, no. 4, pp. 302–311, 2012.
[7] B. Parashar, A. Naithani, P.K. Bhatia, “Analysis of a 3-Unit Induced Draft Fan System with One Warm Standby” *International Journal of Engineering Science and Technology* (IJEST), vol. 4, no.11, pp. 4620-4628, November 2012.
[8] G. Taneja and R. Malhotra, “Cost-benefit analysis of a single unit system with scheduled maintenance and variation in demand,” *Journal of Mathematics and Statistics*, vol. 9, no. 3, pp. 155–160, 2013.
[9] Institution of Railway Signal Engineers minor Railways section guideline on TRAIN DETECTION BONDING AND CABLES, ref no. TC11, issue no. 1.0, March 2013.
[10] Anjali Naithani, Bhupender Parashar, P.K. Bhatia and Gulshan Taneja, “Probabilistic analysis of a 3-unit induced draft fan system with one warm standby with priority to repair of the unit in working state,” *International Journal of System Assurance Engineering and Management*, ISSN 0975-6809, 2017.

An Expolatory Study Software Process Models & Software Development Process

Harish Mahajan
Research Scholar, Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Dr Pushpneel Verma
Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Abstract— **We are facing a period where software study have a huge measurement involving tiny resources, high risk and a wide range of available approaches. In this scenario the Software Development Methodologies (SDMs) could be proved to be a useful ally, but very dangerous and still fatal if misused. The big issue around this matter is how to decide the appropriated SDM that fats a specific study. In the given scope, in this dissertation we are describing a framework for comparing SDMs delivering a position of procedures that shall be followed when the choice of an SDM are ready. We have to approaches of the framework by applying it to be a group of SDMs that was chosen by their popularity and significance. The identified themes were noted down and compared with the notes made during the interviews to ascertain that the former are indeed a true reflection of the discussion in the interviews. This two-step process also verified that the transcription process had not changed the original data generated in the interviews. Five SPI risks were identified from the interview data: organisational politics, lack of support, lack of defined SPI implementation methodology, lack of awareness and lack of resources that are generally considered critical by Australian practitioners. The results also reveal the similarities and the differences in the risks identified by different group of practitioners (i.e. developers, managers and senior managers), different type of organisations (i.e. small–medium and large) and organisations with mature and immature software development processes.**

Keywords: SDM; Model; Process; Method; Analysis

I. INTRODUCTION

In software engineering, a software development process is the process of dividing software development work into distinct phases to improve design, product management, and project management. ... For example, there are many specific software development processes that fit the spiral life-cycle model.

In software engineering, a software development process is the process of dividing software development work into distinct phases to improve design, product management, and project management. It is also known as a software development life cycle(SDLC). The methodology may include the pre-definition of specific deliverables and artifacts that are created and completed by a project team to develop or maintain an application.[1]

Most modern development processes can be vaguely described as agile. Other methodologies include waterfall, prototyping, iterative and incremental development, spiral development, rapid application development, and extreme programming.

Some people[who?] consider a life-cycle "model" a more general term for a category of methodologies and a software development "process" a more specific term to refer to a specific process chosen by a specific organization. For example, there are many specific software development processes that fit the spiral life-cycle model. The field is often considered a subset of the systems development life cycle.

Other high-level software project methodologies include:

Behavior-driven development and business process management[14]

Chaos model - The main rule is always resolve the most important issue first.

Incremental funding methodology - an iterative approach

Lightweight methodology - a general term for methods that only have a few rules and practices

Structured systems analysis and design method - a specific version of waterfall

Slow programming, as part of the larger Slow Movement, emphasizes careful and gradual work without (or minimal) time pressures. Slow programming aims to avoid bugs and overly quick release schedules.

V-Model (software development) - an extension of the waterfall model

Unified Process (UP) is an iterative software development methodology framework, based on Unified Modeling Language (UML). UP organizes the development of software into four phases, each consisting of one or more executable iterations of the software at that stage of development: inception, elaboration, construction, and guidelines. Many tools and products exist to facilitate UP implementation. One of the more popular versions of UP is the Rational Unified Process (RUP).

II. PROCESS META-MODELS

Some "process models" are abstract descriptions for evaluating, comparing, and improving the specific process adopted by an organization.

ISO/IEC 12207 is the international standard describing the method to select, implement, and monitor the life cycle for software.

The Capability Maturity Model Integration (CMMI) is one of the leading models and based on best practice. Independent assessments grade organizations on how well they follow their defined processes, not on the quality of those processes or the software produced. CMMI has replaced CMM.

ISO 9000 describes standards for a formally organized process to manufacture a product and the methods of managing and monitoring progress. Although the standard was originally created for the manufacturing sector, ISO 9000 standards have been applied to software development as well. Like CMMI, certification with ISO 9000 does not guarantee the quality of the end result, only that formalized business processes have been followed.

ISO/IEC 15504 Information technology — Process assessment also known as Software Process Improvement Capability Determination (SPICE), is a "framework for the assessment of software processes". This standard is aimed at setting out a clear model for process comparison. SPICE is used much like CMMI. It models processes to manage, control, guide and monitor software development. This model is then used to measure what a development organization or project team actually does during software development. This information is analyzed to identify weaknesses and drive improvement. It also identifies strengths that can be continued or integrated into common practice for that organization or team.

ISO/IEC 24744 Software Engineering — Metamodel for Development Methodologies, is a powertype-based metamodel for software development methodologies.

SPEM 2.0 by the Object Management Group

Soft systems methodology - a general method for improving management processes

Method engineering - a general method for improving information system processes

III. SOFTWARE PROCESS

A software process (also knows as software methodology) is a set of related activities that leads to the production of the software. These activities may involve the development of the software from the scratch, or, modifying an existing system.

Any software process must include the following four activities:

Software specification (or requirements engineering): Define the main functionalities of the software and the constrains around them.

Software design and implementation: The software is to be designed and programmed.

Software verification and validation: The software must conforms to it's specification and meets the customer needs.

Software evolution (software maintenance): The software is being modified to meet customer and market requirements changes.

IV. SOFTWARE PROCESS MODELS

A software process model is a simplified representation of a software process. Each model represents a process from a specific perspective.

We're going to take a quick glance about very general process models. These generic models are abstractions of the process that can be used to explain different approaches to the software development. They can be adapted and extended to create more specific processes.

Some methodologies are sometimes known as **software development life cycle** (SDLC) methodologies, though this term could also be used more generally to refer to any methodology.

V. SOFTWARE DEVELOPMENT

Problem solving in software development consists of the following activities:
i. Understanding the problem
ii. Deciding a plan for the solution
iii. Coding the planned solution
iv. Testing the actual program [2]
These activities may be very complex for large systems. So, each of the activity has to be broken into smaller sub-activities or steps. These steps are then handled effectively to produce a software project or system. The basic steps involved in software project development are:
i. Requirement analysis
ii. Design
iii. Coding
iv. Testing
In addition, there is a fifth step, ―maintenance‖ that consists of maintaining the system after deployment, i.e. delivery to the customer. Unlike hardware, software does not wear out. But, it is very likely that some errors of the system, which were not found during the software testing phase, may be found by the customer. These errors or bugs need to be reported and resolved immediately. Also, over time, as newer technologies and platforms are developed, system starts becoming outdated. It is important to provide new features to the system after intervals and make it compatible with various latest platforms.

Incremental Vs Waterfall Model

Incremental software development is better than a waterfall approach for most business, e-commerce, and personal systems.
By developing the software incrementally, it is cheaper and easier to make changes in the software as it is being developed.

Compared to the waterfall model, incremental development has three important benefits:

1. The **cost of accommodating changing** customer requirements is reduced. The amount of analysis and documentation that has to be redone is much less than that's required with waterfall model.
2. It's easier to get **customer feedback** on the work done during development than when the system is fully developed, tested, and delivered.
3. More **rapid delivery** of *useful* software is possible even if all the functionality hasn't been included. Customers are able to use and gain value from the software earlier than it's possible with the waterfall model.

VI. Conclusion

It's useful when the requirements are clear, or following a very structured process as in critical systems which needs a detailed, precise, and accurate documents describes the system to be produced.

Not good when requirements are ambiguous, and doesn't support frequent interaction with the customers for feedback and proposing changes. It's not suitable for large projects that might take long time to be developed and delivered.

Prototype

Again, it's an early sample, or release of a product built to test a concept or to act as a thing to be replicated or learned from.

This is very useful when requirements aren't clear, and the interactions with the customer and experimenting an initial version of the software results in high satisfaction and a clearance of what to be implemented.

It's downsides are, good tools need to be acquired for quick development (like coding) in order to complete a prototype. In addition, the costs for for training the development team on prototyping may be high.

They're suited for large projects, less expensive to the change of requirements as they support customer interactions with each increment.

Initial versions of the software are produced early, which facilitates customer evaluation and feedback.

They don't fit into small projects, or projects that waterfall are best suited for; A structured process with a detailed, and accurate description of the system.

It's good for high risky or large projects where the requirements are ambiguous. The risks might be due to cost, schedule, performance, user interfaces, etc.

Risk analysis requires highly specific expertise, and project's success is highly dependent on the risk analysis phase. It doesn't work well for smaller projects.

It suits small-medium size project, with rapidly changes in the requirements as customer is involved during each phase.

Very limited planning is required to get started with the project. It helps the company in saving time and money (as result of customer physical interaction in each phase). The daily meetings make it possible to measure productivity.

Difficult to scale up to large projects where documentation is essential. A highly skilled team is also needed.

If team members aren't committed, the project will either never complete or fail. And there's always a limitation in time, like in increments, meetings, etc.

The four basic process activities of specification, development, validation, and evolution are organized differently in different development processes.

In the waterfall model, they are organized in sequence, while in incremental development they are interleaved. How these activities are performed might depend on the type of software, people involved in development, etc.

References

[1]. Baddoo, N. & Hall, T., 2002. Motivators of Software Process Improvement: An analysis of practitioners' views. *Journal of Systems and Software*, 62(2), pp.85–96.

[2]. Basri, S. & O'Connor, R. V., 2010. Understanding the perception of very small software companies towards the adoption of process standards. In *European Conference on Software Process Improvement*. Springer, pp. 153–164.

[3]. Beck, K., 2000. *Extreme programming explained: embrace change*, Addison-Wesley Professional.

[4]. Bucero, A., 2007. Failing to succeed. *PM Network*, 21(11), p.21. Available at: http://www.pmi.org/learning/library/negative-positive-project-failure-4577 [Accessed April 20, 2017].

[5]. Cramm, S., 2001. Managing Software Development Efficiently. *CIO*. Available at: http://www.cio.com/article/2441202/project-management/managing-software-developmentefficiently. html [Accessed April 20, 2017].

[6]. DeMarco, T. & Lister, T., 2013. *Peopleware: productive projects and teams*, Addison-Wesley.

[7]. Deterding, S. et al., 2011. From game design elements to gamefulness. *Proceedings of the 15th International Academic MindTrek Conference on Envisioning Future Media Environments - MindTrek '11*, pp.9–11.

[8]. Dubois, D.J. & Tamburrelli, G., 2013. Understanding gamification mechanisms for software development. In *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*. New York: ACM, p. 659.

Identification of Database Intrusion Detection Based System on Unpopular Commands

Shuaib Anwar
Research Scholar, Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Dr. Raghav Mehra
Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Abstract— **This abstract proposes a mechanism that enables simultaneous detection of malicious information access through the web analysis of the direction Systems (DBMS). The planned mechanism uses a directed graph representing the profile of valid transactions to notice criminal accesses to information, that area unit seen as unauthorized sequences of Structured Query Language (SQL) commands. The dissertation proposes a standard rule that learns the chart representing the outline of the transactions dead by the users. This mechanism want to defend ancient information applications from information attacks further as internet primarily based applications from SQL injection kinds of attacks. The planned mechanism is generic and may be employed in most business database management system, adding simultaneous detection of malicious information access to classical information security mechanisms.**

In computer intrusion detection one attempts to identify unauthorized accesses to computer accounts. There are two main approaches to intrusion detection: pattern recognition and anomaly detection. Pattern recognition is the attempt to recognize general patterns in command usage that stem from known attacks such as exploiting a software bug. The approach has the disadvantage that it cannot defend against previously unknown software bugs, or any unauthorized user with the knowledge of the account password. Anomaly detection, on the other hand, attempts to identify an unauthorized user by identifying unusual, for the account holder, usage of the computer.

Keywords: SQl; Plan; Process; Users; Malicious

I. INTRODUCTION

In external attack unauthorized attempts are taken place to access or destroy private data and in insider attack the malicious actions are executed by authorized users. The protection of database by the use of encryption techniques, where the database may be encrypted but this kind of system may lead to degradation of query performance. However, in connection with detection of database intrusions few appropriate mechanisms have been proposed in [4, 5]. This paper proposes an innovative mechanism for the detection of database intrusions in DBMS. The proposed approach is also extended to incorporate the CBF. This approach is considered as transaction level approach and is used to detect the malicious transactions in the database.

The profiles computed can be used to detect misuse behaviour in particular insides abuse. The main drawback of this approach is that it has only been described theoretically, and no empirical evidence has been presented. Lee et al. [7] have proposed a real-time database intrusion detection using time signatures. Real-time database systems have a deal with data that changes its value with time. This intrusion detection model observes the database behavior at the level of sensor transaction. If a transaction attempts to update a temporal data which has already been updated in that period, an alarm is raised. Wenhui et al. [8] proposed a two-layer mechanism to detect intrusions against a web-based database services. However, they have not used different level of granularity or intra-transactional and inter-transactional features in their model. Hu et al. [9] determine the dependency among data items where data dependency refers to the access correlations among data items. These data dependency are generated in the form of association rules. Transactions that do not follow any of the mined data dependency rules are marked as malicious transactions. In this paper equal important are given to the each attributes and there is no concept of attribute sensitivity. The attributes sensitivity problem addresses by Srivastava et al. [10], where some of the attributes are considered more sensitive to malicious modification compared to others. They suggest a weighted data mining algorithm for finding dependencies among sensitive attributes. The basic history about the log file is that it consists the information about the committed transactions those are executed in the secure environment by the authorised users. Profile creator takes these offline audit log data as input and generates the transactions profile and these transactions profile are considered as authorised profiles and stored at the system, after that these authorised transactions profile are used at the detection phase. Therefore any database application user wants to execute any transaction in DBMS then he/she will submit the raw queries to feature selector. Feature selector extracts the required features like command type, target object from online raw queries submitted by the users and store it in the online audit trail table. Now detection engine first generate the transaction profile for the data stored in online audit trail and this profile is compared with authorised transactions profile. If online transaction profile matches with authorised transaction profile then the detection engine allows the particular executable transaction to commit into the DBMS. If online transaction profile does not match with authorised transactions profile then the detection engine will never allow the particular transaction to commit into the DBMS and marked as a malicious and system raised the alarm.

II. INTRUSION DETECTION SYSTEMS

Intrusion can be defined as any kind of unauthorised activities that cause damage to an information system. This means any attack that could pose a possible threat to the information confidentiality, integrity or availability will be considered an intrusion. For example, activities that would make the computer services unresponsive to legitimate users are considered an intrusion. An IDS is a software or hardware system that identifies malicious actions on computer systems in order to allow for system security to be maintained (Liao et al., 2013a). The goal of an IDS is to identify different kinds of malicious network traffic and computer usage, which cannot be identified by a traditional firewall. This is vital to achieving high protection against actions that compromise the availability, integrity, or confidentiality of computer systems. IDS systems can be broadly categorized into two groups: Signature-based Intrusion Detection System (SIDS) and Anomaly-based Intrusion Detection System (AIDS).

III. INTRUSION DATA SOURCES

IDS can also be classified based on the input data sources used to detect abnormal activities. In terms of data sources, there are generally two types of IDS technologies, namely Host-based IDS (HIDS) and Network-based IDS (NIDS). HIDS inspect data that originates from the host system and audit sources, such as operating system, window server logs, firewalls logs, application system audits, or database logs. HIDS can detect insider attacks that do not involve network traffic (Creech & Hu, 2014a).

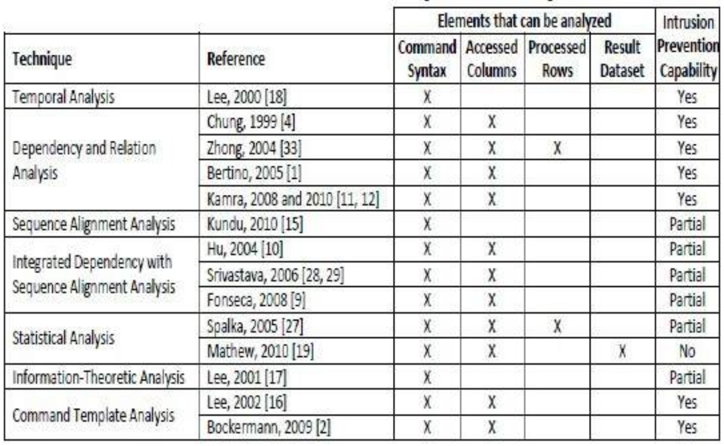
NIDS monitors the network traffic that is extracted from a network through packet capture, NetFlow, and other network data sources. Network-based IDS can be used to monitor many computers that are joined to a network. NIDS is able to monitor the external malicious activities that could be initiated from an external threat at an earlier phase, before the threats spread to another computer system. On the other hand, NIDSs have limited ability to inspect all data in a high bandwidth network because of the volume of data passing through modern high-speed communication networks (Bhuyan et al., 2014).

IV. APPLICATION OF INTRUSION DETECTION IN DATABASES

The applicability of DIDS in database systems depends on the type of environment in which they are supposed to operate. Understanding the characteristics inherent to the typical workloads of each type of environment is critical to determine which type or class of Intrusion Detection (ID) techniques can be more efficient given the nature of those workloads and thus, be considered as more adequate for the specific database system.

1 Transactional versus Analytical Database Systems

Table 1. Database intrusion detection techniques and their coverage



		Elements that can be analyzed				Intrusion
Technique	Reference	Command Syntax	Accessed Columns	Processed Rows	Result Dataset	Prevention Capability
Temporal Analysis	Lee, 2000 [18]	X				Yes
Dependency and Relation Analysis	Chung, 1999 [4]	X	X			Yes
	Zhong, 2004 [33]	X	X	X		Yes
	Bertino, 2005 [1]	X	X			Yes
	Kamra, 2008 and 2010 [11, 12]	X	X			Yes
Sequence Alignment Analysis	Kundu, 2010 [15]	X				Partial
Integrated Dependency with Sequence Alignment Analysis	Hu, 2004 [10]	X	X			Partial
	Srivastava, 2006 [28, 29]	X	X			Partial
	Fonseca, 2008 [9]	X	X			Partial
Statistical Analysis	Spalka, 2005 [27]	X	X	X		Partial
	Mathew, 2010 [19]	X	X		X	No
Information-Theoretic Analysis	Lee, 2001 [17]	X				Partial
Command Template Analysis	Lee, 2002 [16]	X	X			Yes
	Bockermann, 2009 [2]	X	X			Yes

V. DETECTION OF MALICIOUS TRANSACTIONS IN DBMS

In the present work we are particularly interested in the first two types of attacks, which in practice correspond to malicious transactions executed by authorized users or by unauthorized users that gain access to the database by exploring system vulnerabilities. As database security mechanisms are not design to primarily detect intrusions (are designed to avoid the intruder's access to database data), there are many cases where the execution of malicious sequences of SQL1 commands (transactions) cannot be detected (or avoided). The following points present some examples [44,45,46]:

- The DBA does not activate the necessary security mechanisms (e.g., authentication, user privileges, data encryption, auditing, etc), which allows intruders to get access to database data.
- The security mechanisms available are incorrectly configured permitting potential intruders (hackers) to access the database [47].
- Hidden flaws in the database implementation may allow hackers to connect to the database server by exploring those defects.
- Unauthorized users "still" the credentials of authorized users in order to access the database server [48].
- Authorized users take advantage of their privileges to maliciously access or destroy data.

In these situations, such as in many others, manual supervision and manual audit procedures are the only tools that DBAs can use to detect potential intrusion. Malicious transactions may damage the database integrity and availability [49]. However, in spite of the pertinence of the detection of malicious database transactions, the reality is that no practical mechanism able to identify users executing malicious transactions has been proposed so far. This thesis proposes a new mechanism for the detection of malicious transactions in DBMS. As in a typical database environment it is possible to define the profile (sequence of SQL commands) of all the transactions that each user is allowed to execute, the proposed mechanism (named DBMTD - Database Malicious

Transactions Detector) uses that profile of valid transactions to identify user's attempts 1 SQL stands for Structured Query Language, the relational language used by relational DBMS [9] to execute invalid sequences of commands.

VI. Attacks and Control Methods in Database Security

In today's world, many organizations or enterprises have a huge amount of databases to store its data. However, this data storage, maintenance and access are very hard and important issues for organizations. In these organizations, there are huge no of employees wants to use their database from their respective departments. In this case, each user has their own identity proof as like their userID. So by identifying their userID we detect as a person is authorized person or unauthorized person. But this detection is like identify from user name and password so that known as external attack detection. But if in case legitimate or authorized person or employee doing activities that are not fitted for their role & it will be harmful for their organization, then how we decide that those employee activities is suspicious activity & how we detect that, this is known as an inner attack. So data should be stored and accessed by only authorized or legitimate users otherwise this work will be acts as an attack on database. So databases should maintain its own security policies for their safe and secure environment [1]. Gartner research presented that Database transaction's activities and behaviors are examined for to detection of data leaks as well as for detection of malicious insider attacks done by legitimate user or authorized user [2],[3] SQL injection and Data exfiltration attacks are database-related attacks this is not issuing regarding operating system or the network [11]. In database security, inner attacks are done by a trusted people within that organization, so each and every organization should have their own security policy solutions or approaches to fix that problem. Inner attack is mainly detected and resolved by using three techniques, that is Log examining, Query clustering & Policy-based mechanism. These three techniques are performed as like anomaly detection technique in the intrusion-detection system. i.e firstly, observe behavior of each user during training phase, and collect some data and recorded as a benchmark for testing phase.

VII. User Identification /Authentication

A basic security demand is that you just should grasp your users. You want to determine them before you'll confirm their privileges and access rights, then that you just will audit their actions upon the info. User is often attested in some ways before they're allowed to make info. Info authentication includes each identification and authentication of users. External authentication is often performed by the software package or network service. Conjointly the user authentications are often outlined by Secure Socket Layer (SSL), through enterprise roles, through middle tier server authentication conjointly referred to as proxy authentication. This is often the terribly basic demand to make sure security since the identification method defines a collection of individuals that areallowed to access data. To make sure security, the identity is attested and it keeps the sensitive data secure and from being changed by unauthorized user. offender will take totally different approaches like bypass authentication, Default positive identification, privilege step-up, positive identification estimate by brute force and rainbow attack once they conceive to compromise user identification and authentication [1].

VIII. Database Intrusion Detection

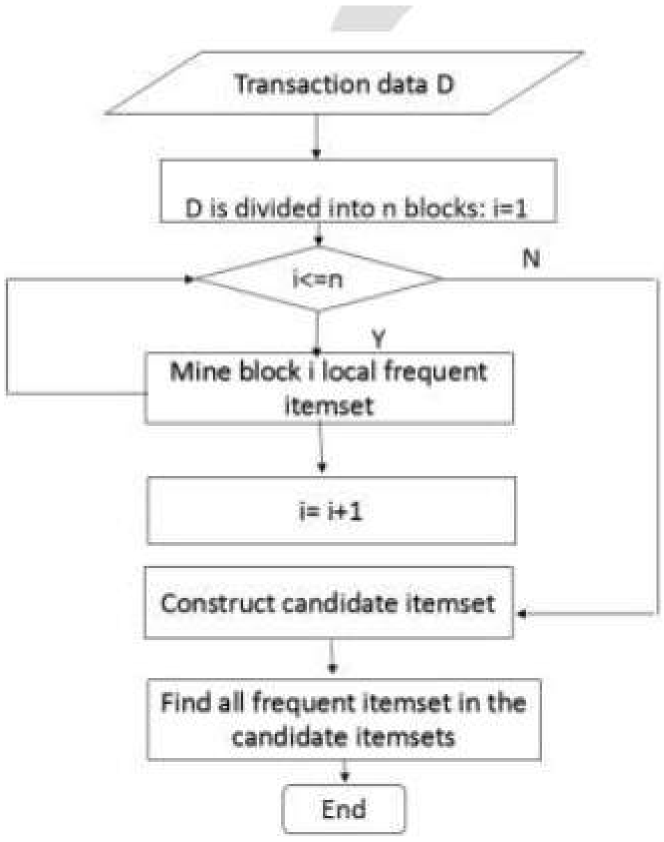
Command level data consisting of user names and commands (without arguments) were generated from output of the UNIX acct auditing mechanism. For illustration purposes a few data points are given in Table 1. Our platform is an SGI server running IRIX 6.2. Some commands recorded by the system are implicitly generated, rather than explicitly typed by the user. For example, each execution of the .profile file or a make file generates commands contained in these files that are also recorded in the data stream.

The first 10,000 commands for each of 53 users were recorded. For the analysis, we separated the commands into training data sets of 5000 commands and test data sets of 1000 commands. For some of the analysis reported below, we split each test data set into 10 replications of 100 commands, and into 2 replications of 500 commands. The use of further information such as time stamps or command arguments is also of interest, but was not considered in this investigation.

Table 1
Example of data used

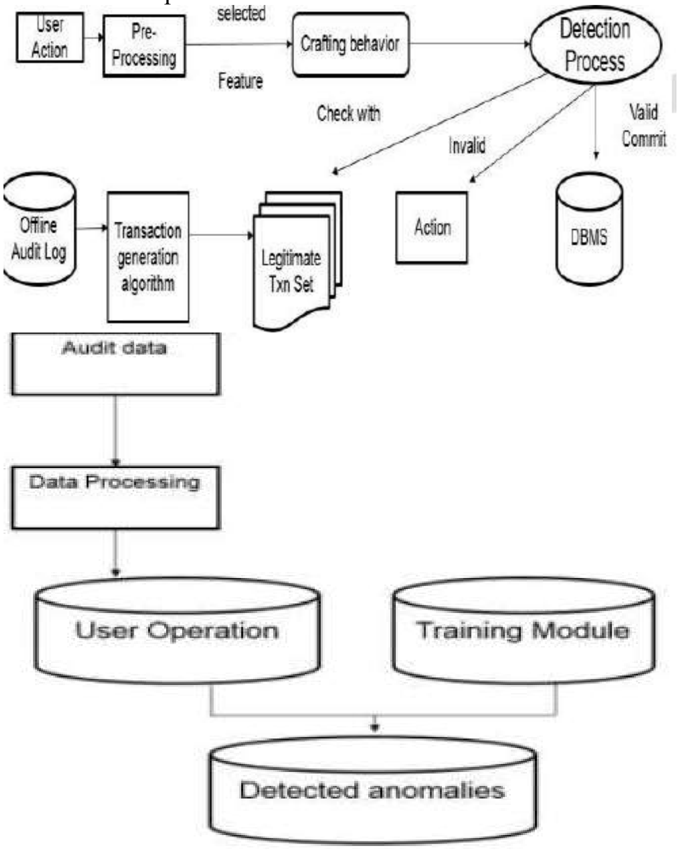
Command name	User
chmod	matt
more	karr
cat	vardi
whoami	theus
sendmail	karr

Database Intrusion Detection based on Improved Association Rule Algorithm [1] It presents an improved association rule algorithm, based on which it builds a database intrusion detection system on the basis of association rules. This system is a circularl and dynamically updated system, but it must first create a set of legitimate access rules from a static database as the basis for the system's judgment. It must repeatedly scan the transaction database to carry out pattern matching for the candidate item sets. Just because the above two flaws, it presents the technology based on the data partition to improve the adaptability and efficiency of the Apriori algorithm. It can use data partition technique for mining frequent item sets with only two times of the whole database scan. As shown in figure, it contains two main processing stages. The first phase, the algorithm will divide the transactional database D into n independent parts For each division (part), to mine all the frequent item sets in which, they are called local frequent item sets. In terms of the whole database D, a local frequent item set is not necessarily the global frequent item set, but any global frequent item set will certainly occur in the local frequent item sets obtained by the partition. This is very easy to get evidence to the contrary. Therefore, the local frequent item sets mined from n partitions can be as the candidate item set of the frequent item sets in the whole database D; and in the second stage again scanning the entire database for the support frequency of all candidate item sets, to finally confirm the global frequent item set. The partition size and number has the standard that each partition can be entirely placed into memory, so each stage only needs to read the database content once, and the entire mining needs to scan the entire database twice.


Transaction data D
D is divided into n blocks: i=1
i<=n
N
Y
Mine block i local frequent itemset
i= i+1
Construct candidate itemset
Find all frequent itemset in the candidate itemsets
End

IX. Database Intrusion Detection by Transaction Signature

The method evaluated here is located on the level of database management system .It focuses on security policies permitted on database system, it is designed to mine audit log of legitimate transaction performed with database and generate signature for legal transactions. The transaction which does not match the signature are declared a malicious transaction according to the policies defined. False positives are valid transactions identified as malicious transactions. In this mechanism the existence of false positives depends on how complete the definition of authorized transaction is. The proposed approach is based on using transaction signature and has learning, detection and response phase. Very briefly, the behaviour of database transaction is collected as a first step to feed the learning phase. Once the database utilization signatures is established, the behaviour learned from audit data is used to concurrently detect database intrusions in detection phase. For intrusive behaviour, this mechanism will alert database administrator. The central theme of my approach will be to learn and create signature from the collected audit data. A basic foundation for intrusion detection is collecting various normal behaviours of Database describe our architecture model in three phases



X. Conclusion

It has emphasized careful data collection, attention to experimental methodology, and error analysis, the latter of which has led to insights about why some masqueraders are harder to detect than others. Nevertheless, masquerade detection remains among the most challenging of classification problems. Masqueraders are people who impersonate other people on the computer. They usually are insiders with malicious intent trying to hide their identity by impersonating other users. They could also be intruders from outside—although in practice most outside intruders immediately try to gain access to the account of the super-user and therefore are a special case. A computer crime and security survey [2] ranking computer security problems in terms of their estimated financial damage found that unauthorized access by insiders was most damaging, accounting for about one-third of the total loss. The uniqueness method can be used as one of the anomaly detection methods within an intrusion detection system. It is not meant to be a stand alone intrusion detection system by itself. There are various methods for detecting the intrusion but still the intrusive activity occurs and malicious transaction takes place. Due to such intrusion the security of confidential and sensitive data is compromised. There is a scope for an improvement in the detecting methods for intrusive activity in database management system and optimizing the detection rate The future work will be enhance the approaches and to overcome the limitation of the processing power and the data storage issues to handle huge amount of information.

References

[1]. Baddoo, N. & Hall, T., 2002. Motivators of Software Process Improvement: An analysis of practitioners' views. *Journal of Systems and Software*, 62(2), pp.85–96.

[2]. Basri, S. & O'Connor, R. V., 2010. Understanding the perception of very small software companies towards the adoption of process standards. In *European Conference on Software Process Improvement*. Springer, pp. 153–164.

[3]. Beck, K., 2000. *Extreme programming explained: embrace change*, Addison-Wesley Professional.

[4]. Bucero, A., 2007. Failing to succeed. *PM Network*, 21(11), p.21. Available at: http://www.pmi.org/learning/library/negative-positive-project-failure-4577 [Accessed April 20, 2017].

[5]. Cramm, S., 2001. Managing Software Development Efficiently. *CIO*. Available at: http://www.cio.com/article/2441202/project-management/managing-software-developmentefficiently. html [Accessed April 20, 2017].

[6]. DeMarco, T. & Lister, T., 2013. *Peopleware: productive projects and teams*, Addison-Wesley.

[7]. Deterding, S. et al., 2011. From game design elements to gamefulness. *Proceedings of the 15th International Academic MindTrek Conference on Envisioning Future Media Environments - MindTrek '11*, pp.9–11.

[8]. Dubois, D.J. & Tamburrelli, G., 2013. Understanding gamification mechanisms for software development. In *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*. New York: ACM, p. 659.

PERFORMANCE IMPROVEMENT OF ROUTING PROTOCOL FOR VEHICULAR MOBILE AD HOC NETWORKS

Harish Chandra Maurya
Research Scholar, Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Dr Pushpneel Verma
Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

***Abstract*— A Vehicular Ad hoc Network (VANET) consists of vehicles which communicate with each other and exchange data via wireless communication links available between the vehicles which are in communication ranges of vehicles to improve the road safety in city. The communication between vehicles is used to provide road safety, comfort and entertainment. VANET consists of a collection of wireless mobile nodes that are dynamically connected and can communicate with each other without the requirement for any pre-existing or centralized infrastructure. There is no static infrastructure. Vehicular Ad hoc networks (VANETs) are a special type of mobile ad hoc networks in which vehicles are simulated as mobile nodes. VANET contains two accesses: access points and vehicles, the access points are fixed and generally connected to the internet. Each protocol has different behavior in relation to performance metrics considered, including the rate of routing packets sent, delay, and the debit. The VANETs (Vehicular Ad hoc Networks) are a new form of mobile ad hoc networks used to establish communications between vehicles or with infrastructure located on roadsides. In a VANET network routing is an important mechanism for finding and maintaining a communication path between a pair of remote nodes.**

Keywords: VANET; Network; Path; Packet; Nodes

I. INTRODUCTION

The main aim of this paper was to identify which ad hoc routing protocol has better performance in highly mobile environment of vehicular ad-hoc network and to give a survey of the VANETs routing Scenario, Vehicular ad hoc networks (VANETs), the existing VANET routing protocols and the exiting two mobility Models. VANETs, investigates different routing schemes that have been developed for VANETs, and providing classifications of VANET routing protocols within two classification forms and gives summarization.

The exchange of information between communicating vehicles without any fixed infrastructure like access points or base stations is an intensive field of research. Upcoming technologies like Fleet Net [1] are based on multi-hop ad hoc networks using Dedicated Short Range Communication for vehicular communications systems. Since each network node acts as wireless station and mobile router at the same time, distant vehicles can communicate with each other using intermediate vehicles for packet forwarding. The application range of such networks may cover safety related applications like the warning of drivers about accidents or congestions. For these purposes, not only vehicles, but also traffic signs may take part in the VANET. Moreover, vehicles may also be provided with Internet access via the ad hoc network using .g. gateways installed along the roadside [2]. The routing of data packets through the VANET is very complex since the network topology and the communication conditions may vary heavily. Several factors like the type of the road, daytime, weather, traffic density and even the driver himself affect the movements of vehicles on a road. Hence, the network topology changes frequently, and the routing protocol used has to adapt it self to theses changes continuously. Up to now, most general work on the performance of routing protocols in MANETs[3], [4] considers only a quite small number of nodes and/or a low mobility as well as simple movement patterns. Thus, one of our goals is to model more realistic movement patterns for the simulations that reflect the movement of vehicles in typical traffic situations. Current routing protocols can be categorized into topology based and location-based protocols. We focus on topology-based routing protocols since these do without any location services for determining the nodes' geographic positions. We chose four protocols to be compared: Ad Hoc on Demand Distance Vector (AODV) [5], Dynamic Source Routing (DSR) [6], Fisheye State Routing (FSR) [7] and Temporally-Ordered Routing Algorithm (TORA) [8]. AODV,DSR and TORA belong to the class of reactive (on-demand)routing protocols that discover routes through the network when they are needed, while proactive routing protocols like FSR continuously maintain routes to all possible destinations.

VANET consists of a collection of wireless mobile nodes that are dynamically connected and can communicate with each other without the requirement for any pre-existing or centralized infrastructure. There is no static infrastructure. Vehicular Ad hoc networks (VANETs) are a special type of mobile ad hoc networks in which vehicles are simulated as mobile nodes. VANET contains two accesses: access points and vehicles, the access points are fixed and generally connected to the internet [1].VANET addresses the wireless communication between vehicles (V2V), and between vehicles and infrastructure access point (V2I). Vehicle to vehicle communication (V2V) has two types of communication: one hop communication (direct vehicle to vehicle communication), and multi hop communication. VANET also has special characteristics that distinguish it from other mobile ad hoc networks; the most important characteristics are: good mobility, self-organization, distributed communication, road pattern restrictions, all these characteristics made VANETs environment a challenging for developing efficient routing protocols. Routing protocols have been built for VANETs environment, which can be classified in further ways, according to different aspects; such as:

protocols characteristics, techniques used, routing information, and so on. There are some classified VANETs routing protocols into five classes: topology-based, position-based, geocast based, broadcast, and cluster based routing protocols, this classification is based on the routing protocols characteristics and techniques used [2], [3], [4]. Moreover, they can be categorized into two classes according to routing strategies: proactive and reactive [8].

II. Research Objectives and Approach

1. To identify ad hoc routing protocol has better performance in highly mobile vehicular ad-hoc network.
2. To study and analyze the selected routing protocol behavior while disseminating data between inter-vehicle communications, an approach of density formulation among traffic flow is used.
3. To improve the overall safety of vehicular traffic, promising traffic management solutions.

III. The Performance Metrics of Ad Hoc Protocols

The performance metrics of ad hoc protocols have been discussed in the first RFC published by the MANET working group [15], by separating these metric to external metrics measurement (how the performance of the protocol is perceived by mechanisms using routing) and internal measurement (for the same level of external performance, two algorithms will deploy different volumes overhead).
External measures the effectiveness of an ad hoc protocol includes:

- The delay
- The debit
- The acquisition time of a road that is only valid for reactive protocols.
- The percentage of segments received out of sequence

To measure the internal efficiency of the protocol, the following metrics are recommended:

- The average number of bits of data transmitted / number of data bits received
- The average number of control bits transmitted / number of data bits received
- The average number of control packets and data transmitted / number of data packets received.

The purpose of this study the performance of ad hoc routing protocols to meet the following three questions:
1) What are the main differences between the ad hoc routing protocols?
2) What routing protocol provides better performance in vehicular ad-hoc networks?
3) What are the factors that influence the performance of these routing protocols?
In attempting to answer the above questions, we used OPNET [14] to compare the performance of the five protocols AODV, DSR, TORA, OLSR and GRP to examine the impact of mobility and density of nodes on the behavior these protocols regarding the speed, the traffic of routing packet sent, the debit and delay.

IV. The Vehicular Networks Are Differentiated by the MANETs

Mobile Ad-hoc Network (MANET) is composed of wireless mobile nodes (e.g. mobile phones, PDAs, Laptops, Pocket PCs, smart car...etc.) interconnected with each other through an autonomous configuration in the absence of any infrastructure. Vehicular ad-hoc network (VANET [1]) is a particular kind of MANET, where smart vehicles act as node (see Figure 1). Indeed, each vehicle is equipped with transmission capabilities to provide communication services among nearby vehicles (inter-vehicle communication V2V) or nearby roadside infrastructures (Vehicle-to-roadside communication V2I) or both combining architectures (see Error! Reference source not found.). This type of network is considered as an enabler for driverless cars of the future, where future networked vehicles are considered as a future convergence of different technologies (computers, communication infrastructures and automobiles) [2], [3]. Though, VANET presents distinctive characteristics; compared to MANET; that offer possibility to increase network performance, but at the same time, they present considerable challenges [4] which hinder the use of a wide range of safety and no safety applications, such as vehicle safety, data transfer, traffic and congestion management, entertainment and providing internet access.

The vehicular networks are differentiated by the MANETs by [5]:

- Highly dynamic topology,
- Frequent disconnection on the network,
- Prediction of the position of the nodes and their movement,
- Communication according to the environment (highway, urban, etc.),
- Delay constraint,
- Unlimited battery power, and
- Computing power (not possible in MANETs).

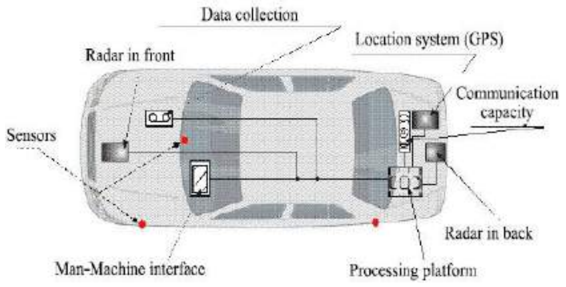


Figure 1: Intelligent vehicle equipped with devices such as computers, network interfaces and sensors that collect information and treat them [6].

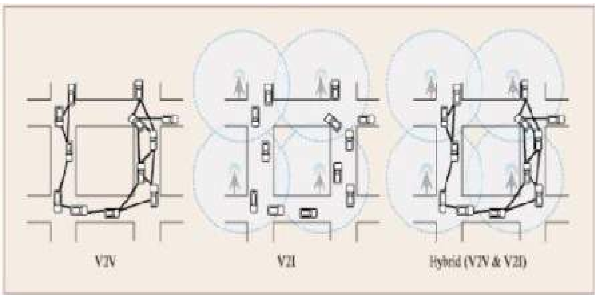


Figure 2: Different modes of existing communications in vehicular ad-hoc networks

The main problematic in Vehicular Ad-hoc Networks is the huge amount of mobility making the network topology incredibly dynamic, the frequent disconnection on the network, and the lack of organization of transmitted data according to the environment in such network. In addition to that, users can receive ambiguous data; these data can not correspond to the field of source of the sender and may lead the receiver to error.

In addition, we used this wide variety of protocols to provide a practical and useful reference, for further study on different classes of vehicular routing protocols to support QoS, or for a future improvement.

The VANETs (Vehicular Ad hoc Networks) [1,2] are a new form of mobile ad hoc networks used to establish communications between vehicles or with infrastructure located on roadsides. They are characterized by a dynamic topology and are generally ad hoc networks that are to say without management and control infrastructure of communication.

V. CURRENT WORK AND PRELIMINARY RESULTS

The main factors are related with the calculations of particular routing metrics. They identify the accumulated results from the output trace files which are generated by the simulator upon their specified inputs from mobility and traffic files. There are various routing metrics devised in different literatures to signify the importance and measuring purposes of numerous routing protocols. Realistic Traces, complete surveys along with the taxonomy of these metrics with their particular classifications. The two highly discussed metrics which are very useful in differentiating the performing trends of routing; and specially picked by similar assessments of such protocols while analyzing ns-2 traces are used for results generation in this research project. They are "Packet Delivery Ratio / Packet Delivery Fraction" and "Average End-to-End Delay".

Packet Delivery Ratio (PDR): It is the fraction of packets generated by received packets. That is, the ratios of packets received at the destination to those of the packets generated by the source. As of relative amount, the usual calculation of this system of measurement is in percentage (%)form. Higher the percentage, more privileged is the routing protocol.

Average End-to-End Delay (E2E Delay): It is the calculation of typical time taken by packet (in average packets) to cover its journey from the source end to the destination end. In other words, it covers all of the potential delays such as route discovery, buffering processes, various in-between queuing stays, etc, during the entire trip of transmission of the packet. The classical unit of this metric is millisecond (ms). For this metric, lower the time taken, more privileged the routing protocol is considered.

VI. WORK PLAN AND IMPLICATIONS

In this report I would like to conclude that I have taken the real city scenario and tried to run 1200 vehicles in an hour and 20,000 to 22,000 in a day and check the results using simulators. We have used different simulators to obtain the results and send the data between inter-vehicle properly according to the incident occurred on road which helps in avoiding accidents and traffic jam and also we have tried to analysis the best routing protocol which will help us to send our data and give better performance. So the results obtain DSR routing protocol using DSRC/IEEE 802.11p gives use better performance results to send our own message according to the event occurred. We have also checked the results on national highway where in a day 24,000 vehicles pass through the express highway and V2V communication using DSRC/IEEE 802.11p and also using DSR routing protocol gives better performance results as compared to other routing protocols. In future we may increase the strength of vehicles in city and on national/express highway and check the results and also provide other information to the vehicle operators (i.e. nearest petrol pumps, hotels etc.)

VII. NEED OF THE PROPOSED RESEARCH WORK

As per our knowledge, no one has compared all the three kinds of TBRPs. To identify the right protocol for VANET environment, we first compared proactive, reactive and hybrid TBRPs. We selected the representative protocols from each category of TBRPs; DSDV [31] from proactive, ZRP [37] from hybrid, and AODV [36] and DSR [33] from the reactive. These protocols have been compared on the basis of some important performance metrics such as routing overhead, delay, throughput and number of dropped packets.NS2 simulator [56] is widely preferred by the networking research community so it has been used for the simulation experiments. DSDV, AODV and DSR protocols are already available into NS2, so experiments can be conducted with them directly. However, for ZRP, a patch has been integrated into the NS2 [84] [85]. In order to perform the simulations, different traffic files have been generated by varying the number of nodes, speed and pause time. After the overall analysis of results, it is concluded that the reliability of AODV and DSR protocols is better than ZRP and DSDV protocols. For the critical performance analysis in a VANET scenario, we have chosen the best TBRPs, AODV and DSR, along with a position based RP LAR. LAR protocol is not supported in NS2, so we requested Tracy Camp [88] for the code and after several modifications, integrated it with NS2. SUMO (Simulation of Urban Mobility) [50] traffic simulator has been used to generate the VANET scenario files with a variation in speed and the number of nodes. For performance evaluation, PDR (Packet Delivery Ratio), routing overhead, throughput and delay metrics have been considered. It has been observed from the results, that for all the metrics except delay, DSR performance is very poor in comparison to the other two protocols. LAR achieves a higher throughput with low routing overhead and less delay in comparison to other two protocols. Especially in a dense city traffic environment, LAR performance is much better than AODV protocol.

References

[1] F. K. Karnadi, Z. H. Mo, and K. c. Lan, “Rapid generation of realistic mobility models for vanet,” in IEEE WCNC, 2007, pp. 2506–2511.

[2] J. H¨arri, F.Filali, and C. Bonnet,”Mobility Models for Vehicular Ad Hoc Networks: A Survey and Taxonomy”, Technical Report RR-06-168, Institut Eurecom, January 2007.

[3] F. Karnadi, Z. Mo, K.-C. Lan, ”Rapid Generation of Realistic Mobility Models for VANET”, in Proc. of the IEEE Wireless Communication and Networking Conference (WCNC’07), March 2007.

[4] A.Mahajan, N. Potnis, K. Gopalan, and A.I.A.Wang, “Urban mobility models for vanets" in Proceedings of the 2nd IEEE International Workshop on Next Generation Wireless Networks,December 2006.

[5] S. Zeadally, et al., "Vehicular Ad Hoc Networks (VANETS): Status, Results, and Challenges," 2010 IEEE.

[6] A. K. Saha and D. B. Johnson, “Modeling mobility for vehicular adhoc networks,” in Proc. of the 2nd ACM International Workshop on Vehicular Ad Hoc Networks (VANET), Oct. 2004. [Online].

[7] Vaishali D. Khairnar and S.N. Pradhan,” Mobility Models for Vehicular Ad-hoc Network Simulation”,in IEEE Symposium on Computers & Informatics,2011, pp.460-465.

[8] M. Tamilarasi, Shyam. Sunder. V. R, U. M.Haputhanthri, C. Somathilaka, N. R. Babu, S.handramathi and T. G. Palanivelu “Scalability Improved DSR Protocol for MANETs”, International Conference on Computational Intelligence and Multimedia Applications 2007, pp. 283-287.

[9] Md. Anisur Rahman, Alex Talevski " Performance Measurement of Various routing Protocols in ad-Hoc Network", IMECS, March 18-20,2009, Hong Kong.

[10] Subir Kumar Sarkar and Basavraju, Ad-hoc-mobile-wireless-networks-principles-protocols-andapplications- Charles E. Perkins, Ad Hoc Networking, Addison-Wesley, March 2005.

[11] Hao-jun Li and Fei-yue Qiu, “Research on Mechanism Optimization of ZRP Cache Information Processing in Mobile Ad Hoc Network”, IEEE-2007

[12] C. Perkins, E. Belding-Royer, and S.Das, Adhoc On-Demand Distance Vector (AODV) Routing, RFC 356,2003.

[13] Rajeswari.M, Dr. P. Uma Meheswari,Bhuvaneshwari.Goweri, Performance analysis of AODV, DSR, TORA and OLSR to achieve group communication in MANET,IEEE-Fourth International Conference on Advanced Computing,MIT, Anna University, Chennai. December 13-15, 2012.

[14] Hui Liu Wei Huang Xu Zhou Wang, X.H"A Comprehensive Comparison of Routing Metrics for WirelessMesh Networks" in Networking, Sensing and Control, ICNSC, IEEE International Conference, April 2008.

[15] T. Clausen and P. Jacquet "Optimized Link State Routing Protocol (OLSR)." RFC 3626, IETF Network Working Group, October 2003.

[16] K.C.Lee, U.Lee, “Survey of Routing Protocols in Vehicular Ad-hoc Networks” 2010.

Cluster-Based Routing Protocol for Effective Way to Prolong the Lifetime of a Wireless Sensor Network (WSN)

Shipra Khandelwal
Research Scholar, Department of CS
Bhagwant University
Ajmer, Rajasthan, India

Dr Pushpneel Verma
Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Abstract— **Several cluster-based routing techniques can be found in [3,4] where periodic cluster head election and re-clustering is performed. In most of these techniques, cluster heads are elected either independently or based on competition of different weight functions. Similar cluster formations are followed by a joining message, either to a nearby cluster head or to a cluster head with a higher residual energy [5–27]. This can balance the energy consumption of the nodes but does not ensure the maximization of the steady state (the time the first node dies in the network) and the network lifetime as well. Because only the residual energy of a node is one of the key criteria to cluster head selection, not considering the average distance from member nodes does not guarantee that the energy consumption of the nodes is saved during intra-cluster communication.**

Many routing protocols have been proposed for WSNs in the last decade [3]. It has been proven that distributed clustering with data-fusion technique is more energy-efficient when compared to other routing protocols [4].

Keywords: Energy; Cluster; Data; Routing; Nodes

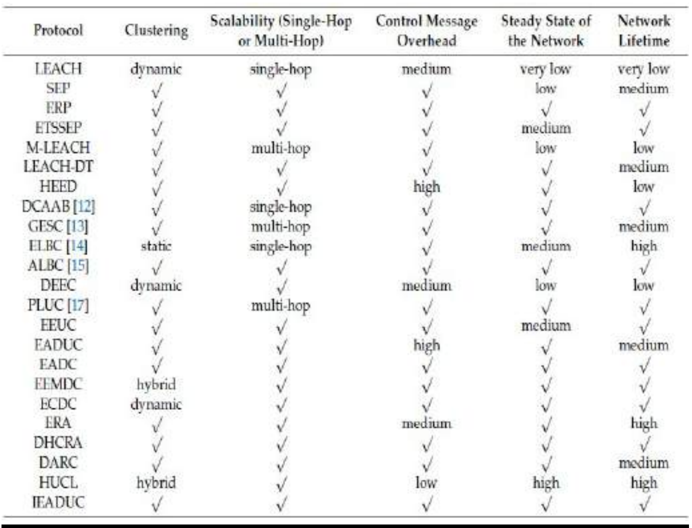
I. INTRODUCTION

Wireless sensor networks (WSNs) consist of sensor nodes with sensing and communication capabilities that have gained enormous attention for their usage in many applications, such as internet of things (IoT) [1,2]. Most wireless sensors are powered with batteries as limited energy sources. In addition, non-rechargeable energy sources strongly require minimizing energy consumption of the nodes and then maximizing network lifetime. Therefore, a lot of research has focused on energy conservation of sensor nodes in long-run operations of WSNs. As a somewhat effective technique, clustering is an efficient way to save energy consumption of the sensor nodes. In the clustering process, sensor nodes are grouped into clusters. A node is selected as a cluster head, a leader in a cluster, and the remaining nodes within the cluster are considered cluster members. Sensor nodes sense the physical parameters related to their environment and send the information to their corresponding cluster heads. Cluster heads then aggregate the data and send it to a remote base station (BS) or sink using single-hop or multi-hop that depends on the distance of the BS. Moreover, periodic cluster-head election and re-clustering processes of the protocols require the broadcasting of control messages in each round, which is the energy waste of the power-limited sensor nodes. One solution is to piggyback the weight value of the nodes with the local data sent to a cluster head to hand over the role [26,27]. Thus, the number of control messages is minimized regarding the cluster-head elections throughout the network lifetime. But, consideration of energy saving in intra-cluster communication of the protocols does not exist yet. In this paper, we propose an energy-centric cluster-based routing protocol called ECCR (energy-centric cluster-based routing) for WSNs that addresses the aforementioned issues. Firstly, we propose predefined static clusters that reduce the control message overhead of the cluster formation issue. Secondly, we introduce a caretaker for cluster-head election technique inspired by L. Malathi et al. [26], where the ranks' information of the nodes are piggybacked along with the local data. A former cluster head from the previous round is responsible to hand over the role to a prospective cluster head in the current round. This reduces the control message overhead regarding the cluster head elections throughout the network lifetime. For data gathering and forwarding, the cluster heads are elected and selected based on the higher rank among the nodes. The rank of a node is defined by the factors which have a major influence on the energy consumption of the node. Experiments are performed on the proposed ECCR to compare with existing protocols such as EADUC, HUCL and IEADUC in [19,26,27], respectively.

The key approach of DARC is to adaptively adjust the routing mode of cluster heads to balance the energy consumption during inter-cluster communication. Even though the periodic clustering can distribute the energy consumption of cluster heads among all nodes, the imbalance in energy consumption exists due to the random location of nodes with respect to the distance between the BS. To address this problem, a relay cluster head is selected among the cluster heads on the basis of relay mode (i.e., CH-L and CH-H denote a cluster head with low and high energy) that is defined by the residual energy of the cluster heads. A CH-H selects the BS as its next hop and acts as a relay node for a CH-L. This process balances the energy consumption among the cluster heads by distributing the relay tasks to the CHs-H, which results in an improved network lifetime. Unlike the typical cluster head election processes of the protocols, hybrid unequal clustering (HUCL) protocol suggests cluster head-hand-over and piggybacking techniques [26]. In the head-hand-over technique, a former cluster head delivers the role of cluster head to an elected node. In the piggybacking technique, the value of weight function is sent along with the local data to an associated cluster head. This reduces the control message overhead during cluster-setup phases.

II. The Metrics and Performance of the Protocols

It combines the cluster head-hand-over technique of HUCL with the same cluster-head election technique of EADUC. Meanwhile, it modifies the relay function of a relay-node selection towards the BS. Unlike the EADUC, a relay node is selected on the basis of relay values defined by three factors; the residual energy of a cluster head, number of member nodes associated with the node, and the costs of data aggregation and transmission to a next hop. This achieves an enhanced network lifetime by contrast with the EADUC and HUCL. IEADUC can prolong the steady state and the network lifetime over the protocols. In all the algorithms, the cluster-head elections are based on different weight functions, where the factors are mostly residual energy of a node, number of neighbor nodes, and distance from the BS. But, in these weight functions, the intra-cluster communication distance has also not taken into consideration, whereas it has a significant impact on the overall energy consumption of the nodes. The cluster-heads election are biased by the time T and competition range of broadcasting a head message. Therefore, a cluster might have a less residual-energy-obtaining node as a cluster head compared to some of the member nodes, i.e., in [19–22,24–27]. On average, a high number of control messages is required to complete a single data transmission to the BS. Although some methods of the protocols are very effective to balance the energy consumption among the nodes, the control messages overhead use an excessive amount of energy of the nodes, which limits the network lifetime as a whole Table 1.1 summarizes the comparison between the protocols on the basis of clustering, scalability, control messages overhead in clustering, and energy efficiency that results in steady state and network lifetime.



Protocol	Clustering	Scalability (Single-Hop or Multi-Hop)	Control Message Overhead	Steady State of the Network	Network Lifetime
LEACH	dynamic	single-hop	medium	very low	very low
SEP	√	√	√	low	medium
ERP	√	√	√	√	√
ETSSEP	√	√	√	medium	√
M-LEACH	√	multi-hop	√	low	low
LEACH-DT	√	√	√	√	medium
HEED	√	√	high	√	low
DCAAB [12]	√	single-hop	√	√	√
GESC [13]	√	multi-hop	√	√	medium
ELBC [14]	static	single-hop	√	medium	high
ALBC [15]	√	√	√	√	√
DEEC	dynamic	√	medium	low	low
PLUC [17]	√	multi-hop	√	√	√
EEUC	√	√	√	medium	√
EADUC	√	√	high	√	medium
EADC	√	√	√	√	√
EEMDC	hybrid	√	√	√	√
ECDC	dynamic	√	√	√	√
ERA	√	√	medium	√	high
DHCRA	√	√	√	√	√
DARC	√	√	√	√	medium
HUCL	hybrid	√	low	high	high
IEADUC	√	√	√	√	√

Table 1.1: The metrics and performance of the protocols (Source: Internet)

III. Energy Consumption Model

The energy consumption is based on the first-order radio dissipation model [5]. The transmitter consumes energy E_{TX} in running the radio electronics and transmission amplifier circuitry. The receiver consumes energy E_{RX} in radio electronics. Depending on the transmission distance d, the free space ϵ_{fs} and multipath fading ϵ_{mp} channel models are used. The energy consumption of transmitting and receiving l-bit data over the distance d is according to Equations (1) and (2), respectively. A cluster head node consumes Eda (nJ/bit/signal) amount of energy in data aggregation and consumes Ecom (nJ/bit/signal) in the processing of additional information of ranks and residual energy of associate nodes. We assume the sensed information by the nodes is highly correlated.

$$E_{TX}(l,d) = \begin{cases} l \times E_{elec} + l \times \varepsilon_{fs} \times d^2 & if\ d \leq d_0 \\ l \times E_{elec} + l \times \varepsilon_{mp} \times d^4 & otherwise \end{cases} \quad (1)$$

$$E_{RX}(l,d) = l \times E_{elec} \quad (2)$$

where E_{elec} is the power consumption for transmitting and receiving circuitry, a signal is amplified that depends on the distance d, and the reference distance

$$d_0 = \sqrt{(\varepsilon_{fs} / \varepsilon_{mp})} = 87 \text{ m}.$$

IV. Protocol Details

The whole operation is divided into periodic rounds. Each round consists of the cluster head election and data transmission phases. Cluster members sense the information from the environment and send the collected data to their cluster head. A cluster head receives and aggregates the data from its member nodes and sends the data towards the BS based on the constructed routing path. Data transmission phase should be longer than cluster-head election phase to reduce the overhead of the protocol and to prolong the network lifetime. Several control and data messages are used. The description of the messages is given in Table 1.2 a flowchart of ECCR operation has been given in Figure 1.1.

Table 1.2: Description of the control and data messages (Source: Internet)

Message	Description
Hello_Msg	Tuple (selfid, clusterid), a control message of nodes' initial information.
Node_Msg	Tuple (selfid, selfrank, selfenergy), a control message of members' information.
Handover_Msg	Tuple (selfid, clusterid, headid), a control message of hand over the role of cluster head to a prospective cluster head in a cluster.
Schedule_Msg	Tuple (schedule, order), a control message for assign the time slot for a member node to send local data to an associate cluster head.
Route_Msg	Tuple (selfid, selfrank, selfenergy, disttoBS), a control message to collect neighbor cluster heads' information.
D_Msg	Tuple (selfid, clusterid, selfrank, selfenergy, *'local data'*), a local data message from a member node to associate cluster head.
AD_Msg	Tuple (selfid, clusterid, nexthopid, *'fused data'*), an aggregated data message from a cluster head to a next hop or BS.

V. Data Aggregation Model

In this work, the infinite compressibility model [5] is used for data aggregation. It is assumed that a cluster head aggregates local data gathered from its members into a single packet of fixed length, irrespective of the number of received packets. The radio of each non-cluster head node is turned off until the allocated transmission time of node. The receiver of a cluster head node should be turned on to receive all the local data from the associate member nodes.

These clustering algorithms are commonly based on rotating the role of cluster heads and re-clustering in every round to prolong the network lifetime. The low energy adaptive clustering hierarchy (LEACH) protocol [5] is a pioneer work available in this category. The cluster head election process of LEACH is periodic and for every round, new cluster heads are elected. Although the LEACH protocol distributes the energy consumption among the nodes equally, it leads to additional routing overhead, resulting in excessive use of limited energy of a cluster head. Besides, the protocol assumes single-hop communication between the nodes and BS. It constructs the network by making it less scalable and is unsuitable for a large scale WSN. LEACH gives birth to many protocols that have been developed in the last few years that improved over the network lifetime. The stable election protocol (SEP) has been proposed in reference [6] for heterogeneous WSN.

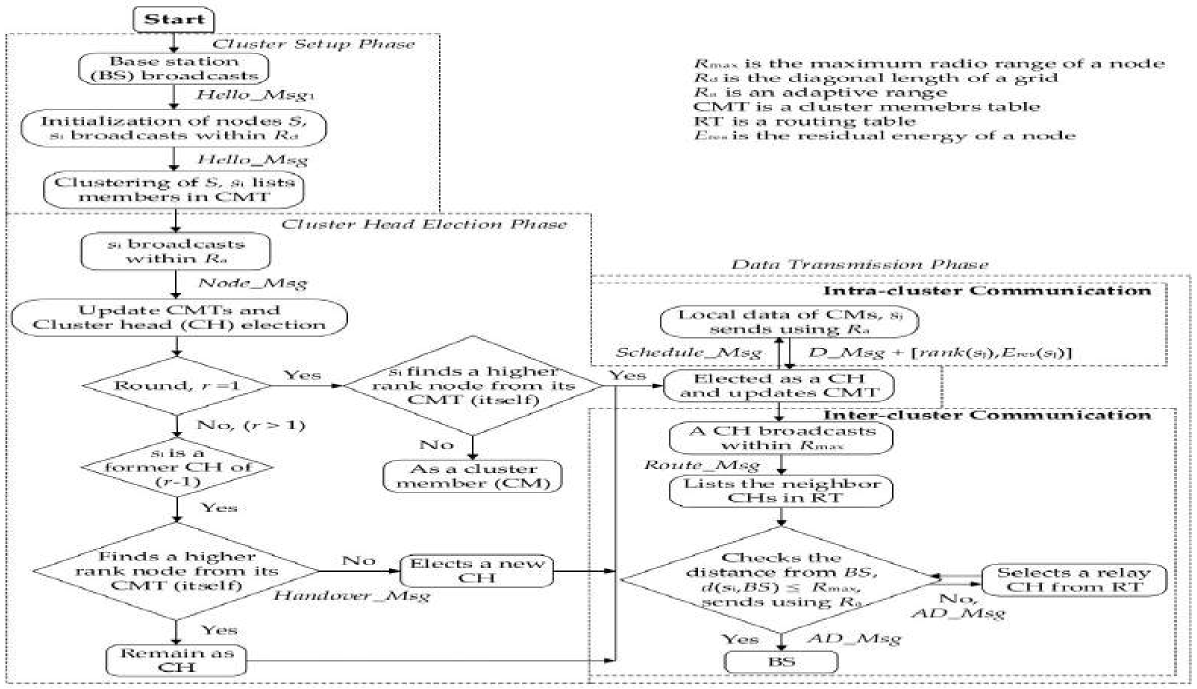

Start
Cluster Setup Phase
Base station (BS) broadcasts
Hello_Msg
Initialization of nodes S, si broadcasts within Rd
Hello_Msg
Clustering of S, si lists members in CMT
Rmax is the maximum radio range of a node
Rd is the diagonal length of a grid
Ra is an adaptive range
CMT is a cluster memebrs table
RT is a routing table
Eres is the residual energy of a node
Cluster Head Election Phase
si broadcasts within Ra
Node_Msg
Update CMTs and Cluster head (CH) election
Round, r =1
Yes
si finds a higher rank node from its CMT (itself)
No
As a cluster member (CM)
No, (r > 1)
si is a former CH of (r-1)
Yes
Finds a higher rank node from its CMT (itself)
No
Elects a new CH
Handover_Msg
Yes
Remain as CH
Data Transmission Phase
Intra-cluster Communication
Local data of CMs, si sends using Ra
Schedule_Msg
D_Msg + [rank(si),Eres(si)]
Yes
Elected as a CH and updates CMT
Inter-cluster Communication
A CH broadcasts within Rmax
Route_Msg
Lists the neighbor CHs in RT
Checks the distance from BS, d(si,BS) ≤ Rmax, sends using Ra
Selects a relay CH from RT
No, AD_Msg
Yes
AD_Msg
BS

Figure 1.1: Flowchart of energy-centric cluster-based routing (ECCR) operation, including cluster setup, cluster-head election and data transmission. (Source: Internet)

References

[1]. Degan Zhang, Guang Li, Ke Zheng, Xuechao Ming, and Zhao-Hua Pan, ”An Energy-Balanced Routing Method Based on Forward-Aware Factor for Wireless Sensor Networks” IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, VOL. 10, NO. 1, FEBRUARY 2014.

[2]. A. Abbasia and M. Younisb , A survey on clustering algorithms for wireless sensor networks, In The International Journal for the Computer and Telecommunications Industry, Vol. 30, No. 14 , 2007, pp. 2826-2841.

[3]. B.Zarei, M.Zeynali and V.Majid Nezhad, Novel Cluster Based Routing Protocol in Wireless Sensor Networks, IJCSI International Journal of Computer Science Issues, Vol.7, Issue 4, No.1, (2010) July, pp.32-36.

[4]. F.Bajaber and I.Awan, Energy ecient clustering protocol to enhance lifetime of wireless sensor network, Journal of Ambient Intelligence and Humanized Computing, Vol. 1, No.4, (2010), pp.239-248.

[5]. W. Heinzelman, A. Chandrakasan and H. Balakrishnan, An applicationspecific protocol architecture for wireless microsensor networks, IEEE Transactions on Wireless Communications, Vol. 1 , Issue 4 , (2002), pp. 660670.

[6]. F. Tang, I.You , S. Guo, M. Guo and Y. Ma, A chain-cluster based routing algorithm for wireless sensor networks, Journal of Intelligent Manufacturing, Published Online, (2010) May 14.

[7]. I.F.Akyildiz, W.Su,Y.Sankarasubramaniam, A Survey on Sensor Networks, IEEE Communications Magazine, 2002, 40(8), pp.102-114.

[8]. W.Heinzelman,A.Chandrakasan,H.Balakrishnan,Energy-Ecient Communication protocol for wireless microsensor networks,in the Proceedings of the 33rd International Conference on System Science(HICSS00), Hawaii, U.S.A., January 2000.

[9]. Stephanie Lindsey and Cauligi S. Raghavendra, PEGASIS: Power-Efficient Gathering in Sensor Information Systems, in Proceedings of the IEEE Aerospace Conference, vol.3,pages 1125-1130, March 2002.

[10]. B.Zarei, M.Zeynali and V.Majid Nezhad, Novel Cluster Based Routing Protocol in Wireless Sensor Networks, IJCSI International Journal of Computer Science Issues, Vol.7, Issue 4, No.1, (2010) July, pp.32-36.

[11]. D. Braginsky and D. Estrin, ”Rumor Routing Algorithm for Sensor Networks,” in the Proceedings of the First Workshop on Sensor Networks and Applications (WSNA), Atlanta, GA, October 2002.

[12]. S. Hedetniemi, S. Hedetniemi, and A. Liestman,”A Survey of Gossiping and Broadcasting in Communication Networks,”Network, vol. 18, 1988.

[13]. Y. Yu, D. Estrin, and R. Govindan, ”Geographical and Energy-Aware Routing: A Recursive Data Dissemination Protocol forWireless Sensor Networks”, UCLA Computer Science Department Technical Report, UCLA-CSD TR-01- 0023, May 2001.

[14]. A. Mohanoor, Data gathering techniques on wireless sensor networks. ProQuest, 2008.

[15]. M. B. Yassein, A. Alzou, Y. Khamayseh, W. Mardini, “Improvement on LEACH protocol of wireless sensor network (VLEACH)”, International Journal of Digital Content: Technology and its Applications, Vol. 3, No. 2, pp. 132–136, 2009

[16]. R. V. Kulkarni, G. K. Venayagamoorthy, “Particle swarm optimization in wireless sensor network: a brief survey”, IEEE Transactions on System, Man, and Cybernetics-Part C: Applications and Reviews, Vol. 41, No. 2, pp. 262-267, 2011

A Study of Search-Based Testing for Embedded System Software

Mohit Mishra
Research Scholar, Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Dr Pushpneel Verma
Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Abstract— **Software testing in very important activity in software development. It is one of the important activity which require lot of time and labour. Different testing techniques are used to find bugs in the software. Testing is involved at different stages of software development like unit testing, integration testing, system testing, acceptance testing etc. Different testing techniques namely dynamic testing, functional testing and structural testing are used to test software. Software Testing is a process of finding errors while executing a program so that we get a zero defect software. It is aimed at evaluating the capability or usability of a program. Software testing is an important means of accessing quality of software. Though a lot of advancements have been done in formal methods and verification techniques, still we need software to be fully tested before it could be handled to the customer side. Thus there are a number of testing techniques and tools made to accomplish the task. Software testing is an important area of research and a lot of development has been made in this field. In this synopsis our aim is to discuss the application of search-based software testing techniques for unit level testing of a real-world telecommunication middleware at Ericsson.**

Search-based test generation for embedded system software units developed using the Function Block Diagrams (FBDs), a graphical language in the IEC 61131-3 standard aimed at programmable logic controllers (PLCs). We consider 279 different components from the train control software developed by Bombardier Transportation, a major rail vehicle manufacturer. The software is compiled into C code with a particular structure. We use a modified hill climbing algorithm for generating test data to maximize MC/DC coverage for assignments with logical expressions in the C code, while retaining the semantics of the original FBD implementation.

Keywords: Task; Level; Data; SoftwareTesting

I. INTRODUCTION

There is an increasing integration of software into many products. This makes software quality a relevant factor in the overall quality of all such products. However, systems engineers and integrators are not software engineering experts and, in particular, have little or no software testing experience. One option to ensure proper quality of the software being integrated is to involve dedicated software engineers to handle software development and testing.

Testing is a process to evaluate the quality of software. It is labour intensive activity. Different types of testing are used to test software. There is scope for automation in the activities of testing but testers experience is very much important for successful testing. Software testing is component of software quality control. Different types of tests are used to testing like unit testing, integration testing, acceptance testing, system testing are used to test a system. Techniques of testing like static testing, functional testing, dynamic testing and structural testing have been used to test the system. We discussed a case study on application of SBST techniques for an industrial embedded software. During the experiments we were able to increase the total branch coverage of existing test cases. So we can say that the employed SBST techniques indicate beneficial results. However, we observed that randomly generated inputs were as effective as more informed search algorithms in many (but not all) cases, which indicate that there are many branches that are easy to cover.

Various difficulties were encountered at the implementation phase due to the specialized execution environment of the embedded software. Therefore, our prototype implementation was highly adapted to the system under test, instead of following a more generic structure. Another important characteristic of the system under test is its complexity and the specialized domain.

II. SEARCH-BASED SOFTWARE TESTING (SBST)

Search-Based Software Testing (SBST) is an excellent fit for the latter option. It has been shown to be a good approach for many different types of testing [3, 2]. It consists of a very generic search component, while those components that are domain-specific are those that systems engineers and domain specialists have their expertise in. These domain-specific components are the representation of the problem and the software as well as the quality criteria used for evaluation. The contribution of this paper, therefore, is to propose a search-based software testing system that will allow domain-specialist users to create test cases for the software they produce, without the need for specialized knowledge of software testing or search-based techniques.

Search-Based Software Testing (SBST) is the application of search techniques to the problem of software testing. SBST has been applied to a variety of testing problems [3, 2], from object-oriented containers [11] to dynamic programming languages [12] and invariants [30]. SBST is a part of the wider area of Search- Based Software Engineering (SBSE), a term coined by Harman and Jones in 2001 [1], but in use earlier, see e.g. [7, 8, 9]. Since search-based approaches have been applied across the software development life-cycle [13], it is reasonable to expect that any conclusions referring to the general area of SBSE can be applied to the particular case of SBST.

The context of the system proposed in this work is that of software testing performed by domain specialists with relatively little knowledge of software testing or search-based techniques. The domain specialists would have extensive knowledge of the capabilities of the system-under-development, its own context and the limitations placed upon it, as well as the quality foci that they would have to pursue for each component of that system. The combination of non-specialist software developers and the importance of domain knowledge and limitations makes it impractical to develop a fitness function up-front. To develop an appropriate fitness function for the component under test, the domain specialist would have to interact with the system and make adjustments to the criteria being used.

III. Running Example

To better illustrate the concepts discussed in this section, we will present an anonymized industrial example. The application we will use is that of a controller enabling a joystick or set of joysticks to handle a mechanical arm. The inputs for the controller software are the joystick signals and sensors that indicate the speed of the basket at the end of the crane. The outputs are the signals to the hydraulic pumps that drive the arm.

The System Under Test (SUT), in this case, is the software for the controller component. The goal of the Search-Based Software Testing System is to generate test cases that ensure the system's compliance to quality standards, ensure that no constraints are broken and discover any additional faults or unexpected behavior.

The Search-Based Software Testing System is the result of applying the methodology presented in this work in the company in question. The system is meant to be tailored for the specific context and company it is expected to function in, yet be general enough to enable domain specialists to test new applications within the confines of that context.

IV. Improving quality

By doing effective testing on software, errors can be minimized and thus quality of software is improved.

For life-critical software like flight control, testing can be much expensive as risk analysis is also involved. Risk analysis means the probability by which a software project can experience undesirable events, such as delays, schedule, outright cancellation and cost overruns and much more. So, a number of test cases and test plans are made in testing which means that the behavior of a program is inspected on a finite set of test cases i.e. inputs, execution preconditions, and also expected outcomes for a particular objective, such as to follow a particular program path or to verify compliance with a specific requirement, for which valued inputs are created. Practically, the set of test cases is considered to be infinite, thus theoretically there are a lot of test cases even for the smallest and simplest program (Stacey, D. A., and Software Testing Techniques). In that case, testing could take a lot of time even months and months to execute. So, how to choose a proper set of test cases? Practically, various techniques are used, and some of them are also correlated with risk analysis, while others are correlated with test engineering expertise.

V. Error Detection

To detect errors a number of testing should be done to make things go wrong to determine if what things should happen when they should not.

Throughout the implementation of this industrial case study we have encountered various problems. We were not familiar with the DSP-C language which prevented us from using various toolboxes available for C, and led us to spend more time on implementation of the code analysis part. Another major implementation work involved the adaptations needed for the execution model of the simulator. This can be seen as a special version of the execution environment problem, which usually refers to the concerns about interacting with the file system, the network, or a database in the context of non-embedded systems [17].

Another major point in the case study is that we did not have enough detailed technical knowledge of the system under test to understand the input space. To overcome this problem, we used existing test cases to automatically craft a test case template. Furthermore, we reduced the input vectors by ignoring the input data structure fields that are not used by any of the existing test cases, which decreases the input size significantly. It is not guaranteed that existing test cases include enough input fields to achieve full branch coverage – in other words, this is a speculative approach. However, results were promising, and we were able to achieve full coverage for most of the FUTs.

VI. Search-Based Testing for Embedded System Software

Search-based test generation for embedded system software units developed using the Function Block Diagrams (FBDs), a graphical language in the IEC 61131-3 standard aimed at programmable logic controllers (PLCs). We consider 279 different components from the train control software developed by Bombardier Transportation, a major rail vehicle manufacturer. The software is compiled into C code with a particular structure. We use a modified hill climbing algorithm for generating test data to maximize MC/DC coverage for assignments with logical expressions in the C code, while retaining the semantics of the original FBD implementation. An experimental evaluation for comparing the effectiveness (coverage rate) and the efficiency (required number of executions) of hill climbing algorithm with random testing is presented. The results show that random testing performs well for most units under test, while around 30% of the artifacts significantly benefit from the hill climbing algorithm. Structural properties of the units that affect the performance of hill climbing and random testing are also discussed.

Embedded software is computer software, written to control machines or devices that are not typically thought of as computers, commonly known as embedded systems. It is typically specialized for the particular hardware that it runs on and has time and memory constraints.[1] This term is sometimes used interchangeably with firmware.

A precise and stable characteristic feature is that no or not all functions of embedded software are initiated/controlled via a human interface, but through machine-interfaces instead.[2]

Manufacturers build embedded software into the electronics of cars, telephones, modems, robots, appliances, toys, security systems, pacemakers, televisions and set-top boxes, and digital watches, for example.[3] This software can be very simple, such as lighting controls running on an 8-bit microcontroller with a few kilobytes of memory with the suitable level of processing complexity determined with a Probably Approximately Correct Computation framework[4] (a methodology based on randomized algorithms), or can become very sophisticated in applications such as airplanes, missiles, and process control systems.[5]

Most consumers are familiar with application software that provide functionality on a computer. Embedded software however is often less visible, but no less complicated. Unlike application software, embedded software has fixed hardware requirements and capabilities, and addition of third-party hardware or software is strictly controlled.

Embedded software needs to include all needed device drivers at manufacturing time, and the device drivers are written for the specific hardware. The software is highly dependent on the CPU and specific chips chosen. Most embedded software engineers have at least a passing knowledge of reading schematics, and reading data sheets for components to determine usage of registers and communication system. Conversion between decimal, hexadecimal and binary is useful as well as using bit manipulation.[7]

Web applications are rarely used, although XML files and other output may be passed to a computer for display. File systems with folders are typically absent as are SQL databases.

Software development requires use of a cross compiler, which runs on a computer but produces executable code for the target device. Debugging requires use of an in-circuit emulator, JTAG or SWD. Software developers often have access to the complete kernel (OS) source code.

Size of the storage memory and RAM can vary significantly. Some systems run in 16 KB of Flash and 4 KB of RAM with a CPU operating at 8 MHz, other systems can rival contemporary computers.[8] These space requirements lead to more work being done in C or embedded C++, instead of C++. Interpreted languages like BASIC (while e.g. Parallax Propeller can use compiled BASIC) and Java (Java ME Embedded 8.3[9] is available for e.g. ARM Cortex-M4, Cortex-M7 microcontrollers and older ARM11 used in Raspberry Pi and Intel Galileo Gen. 2) are not commonly used; while an implementation of the interpreted Python 3 language – MicroPython – is however available expressly for microcontroller use, e.g. 32-bit ARM-based (such as BBC micro:bit) and 16-bit PIC microcontrollers.

VII. Conclusion

A precise and stable characteristic feature is that no or not all functions of embedded software are initiated/controlled via a human interface, but through machine-interfaces instead.[2]

Manufacturers build embedded software into the electronics of cars, telephones, modems, robots, appliances, toys, security systems, pacemakers, televisions and set-top boxes, and digital watches, for example.[3] This software can be very simple, such as lighting controls running on an 8-bit microcontroller with a few kilobytes of memory with the suitable level of processing complexity determined with a Probably Approximately Correct Computation framework[4] (a methodology based on randomized algorithms), or can become very sophisticated in applications such as airplanes, missiles, and process control systems.[5]

The System Under Test (SUT), in this case, is the software for the controller component. The goal of the Search-Based Software Testing System is to generate test cases that ensure the system's compliance to quality standards, ensure that no constraints are broken and discover any additional faults or unexpected behavior.

References

[1]. A. P. Mathur, "Foundation of Software Testing", Pearson/Addison Wesley, 2008.

[2]. IEEE Standard 829-1998, "IEEE Standard for Software Test Documentation" pp.1-52, IEEE Computer Society, 1998.

[3]. D. Gelperin and B. Hetzel, "The Growth of Software Testing", Communications of the ACM, Volume 31 Issue 6, June 1988, pp. 687- 695[history of st]

[4]. D. Richardson, O. O'Malley and C. Tittle, "Approaches to specification-based testing", ACM SIGSOFT Software Engineering Notes, Volume 14 , Issue 9, 1989, pp. 86 – 96[Approaches to specification-based testing]

[5]. S. Rapps and E. J. Weyuker, "Selecting Software Test Data Using Data Flow Information," IEEE Transactions on Software Engineering, April 1985, pp. 367-375

[6]. Harrold Mary Jean, and Gregg Rothermel. "Performing data flow testing on classes." ACM SIGSOFT Software Engineering Notes. Vol. 19. No. 5. ACM, 1994.[acm.pdf]

[7]. Claessen Koen, and John Hughes. "QuickCheck: a lightweight tool for random testing of Haskell programs." Acm sigplan notices 46.4 (2011): 53-64.

[8]. Vilkomir Sergiy A., Kalpesh Kapoor, and Jonathan P. Bowen. "Tolerance of control-flow testing criteria." Computer Software and Applications Conference, 2003. COMPSAC 2003. Proceedings. 27th Annual International. IEEE, 2003.[control ieee]

A Study of LSB Steganography with Colour Image

Annu Sharma
Research Scholar, Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Dr. Amit Kumar Chaturvedi
Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Dr . Kalpana Sharma
Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Abstract— **Steganography is a technique used to hide a message or disguise information from the view of all but other the authentic receiver.The technique involves four stages: secret message preparation, image halftoning, embedding and extraction. Firstly, the secret message is converted from character to a bit-stream. Secondly, a halftone image is created from the cover image in order to determine the embedding pixels In steganography the carrier media include: audio, text, video, and any other type of digital medium [6]. Audio steganography hides messages inside an audio medium. In this type the hidden audio must re- main undetectable; the techniques include Amplitude Modification, Spread Spectrum Coding, Echo Hiding, Phase Coding, and Least Significant Bit (LSB).**

Keywords: LSB; Receiver; Data; SoftwareTesting

I. INTRODUCTION

Steganography practice of concealing a file, message, image, or video within another file, message, image, or video. The word *steganography* combines the Greek words *steganos*, meaning "covered or concealed", and *graphe* (γραφή) meaning "writing". The first recorded use of the term was in 1499 by Johannes Trithemius in his *Steganographia*, a treatise on cryptography and steganography, disguised as a book on magic. Generally, the hidden messages appear to be (or to be part of) something else: images, articles, shopping lists, or some other cover text. For example, the hidden message may be in invisible ink between the visible lines of a private letter. Some implementations of steganography that lack a shared secret are forms of security through obscurity, and key-dependent steganographic schemes adhere to Kerckhoffs's principle.[1]

The advantage of steganography over cryptography alone is that the intended secret message does not attract attention to itself as an object of scrutiny. Plainly visible encrypted messages, no matter how unbreakable they are, arouse interest and may in themselves be incriminating in countries in which encryption is illegal.[2]

Whereas cryptography is the practice of protecting the contents of a message alone, steganography is concerned both with concealing the fact that a secret message is being sent and its contents.

Steganography includes the concealment of information within computer files. In digital steganography, electronic communications may include steganographic coding inside of a transport layer, such as a document file, image file, program or protocol. Media files are ideal for steganographic transmission because of their large size. For example, a sender might start with an innocuous image file and adjust the color of every hundredth pixel to correspond to a letter in the alphabet. The change is so subtle that someone who is not specifically looking for it is unlikely to notice the change.

The first recorded uses of steganography can be traced back to 440 BC when Herodotus mentions two examples in his Histories.[3] Histiaeus sent a message to his vassal, Aristagoras, by shaving the head of his most trusted servant, "marking" the message onto his scalp, then sending him on his way once his hair had regrown, with the instruction, "When thou art come to Miletus, bid Aristagoras shave thy head, and look thereon." Additionally, Demaratus sent a warning about a forthcoming attack to Greece by writing it directly on the wooden backing of a wax tablet before applying its beeswax surface. Wax tablets were in common use then as reusable writing surfaces, sometimes used for shorthand.

In his work Polygraphiae, Johannes Trithemius developed his so-called "Ave-Maria-Cipher" that can hide information in a Latin praise of God. "Auctor Sapientissimus Conseruans Angelica Deferat Nobis Charitas Potentissimi Creatoris" for example contains the concealed word VICIPEDIA.[4]

The stego-image is the final product after a secret message is embedded in the cover object. A secret message will be concealed in a cover-image by applying an embedding algorithm to produce a stego-image. The transmission of the stego-image via a communication channel is performed by a sender to a receiver. To reveal the covert message that is concealed by the sender, the receiver needs to have the destego algorithm which is parameterized by a stego-key to extract the secret message. This is the purpose of a steganographic system where an attacker who does not possess the name of a file or the stego-key for accessing it definitely will not be able to determine whether the file is even present [7]. In an efficient steganographic system, a normal cover medium should not be distinguishable from a stegoobject [6]. Steganography Mechanism Digital images have become commonplace and nowhere are these images more prevalent than on the World Wide Web in the Internet [8]. Using digital images as a carrier medium is suitable for information hiding because of their insensitivity for the human visual system [11]. The vast majority of web pages are impressively sophisticated with colour images and thus Internet users browsing through the web no longer pay attention to sites containing images or to the downloading of images and data files from the Web [8]. Besides, there is a large amount of redundant bits in an image. The redundant bits of an object are those bits that can be altered but the alteration cannot be visibly detected by human eyes [3].

II. Cyber-physical systems/Internet of Things

Academic work since 2012 demonstrated the feasibility of steganography for cyber-physical systems (CPS)/the Internet of Things (IoT). Some techniques of CPS/IoT steganography overlap with network steganography, i.e. hiding data in communication protocols used in CPS/the IoT. However, specific techniques hide data in CPS components. For instance, data can be stored in unused registers of IoT/CPS components and in the states of IoT/CPS actuators.

The cover image which is used here is a color image so, first the three primary colors(R, G, B) are separated. Then the each color is then split into six sub blocks this method is known as blocking. The data hiding is done manually so that in which block the data has to be hidden. If the image size, say 1000x750 pixels (written as width x height by convention), then there would be 1000 columns and 750 rows of data values, or 1000 x 750=750000 pixels total. For 24 bit color, each pixels data contains three 8-bit RGB byte values, or 750, 000x3=2,250,000 bytes. Every pixel is the same size, because a pixel is simply the color of the area between the grid lines. The area will be colored by the one corresponding RGB data value.

III. Least Significant Bits (LSB) insertion

A. Technique basics

Today, when converting an analog image to digital format, we usually choose between three different ways of representing colors:

- 24-bit color: every pixel can have one in 2^24 colors, and these are represented as different quantities of three basic colors: red (R), green (G), blue (B), given by 8 bits (256 values) each.
- 8-bit color: every pixel can have one in 256 (2^8) colors, chosen from a palette, or a table of colors.
- 8-bit gray-scale: every pixel can have one in 256 (2^8) shades of gray.

LSB insertion modifies the LSBs of each color in 24-bit images, or the LSBs of the 8-bit value for 8-bit images.

1) Example:

The letter 'A' has an ASCII code of 65(decimal), which is 1000001 in binary.

It will need three consecutive pixels for a 24-bit image to store an 'A':

Let's say that the pixels before the insertion are:

10000000.10100100.10110101,
10110101.11110011.10110111,
11100111.10110011.00110011

Then their values after the insertion of an 'A' will be:

*1000000**1**.10100100.1011010**0**,*
*1011010**0**.1111001**0**.1011011**0**,*
*1110011**0**.10110011.00110011*

(The values in **bold** are the ones that were modified by the transformation)

The same example for an 8-bit image would have needed 8 pixels:

10000000, 10100100, 10110101, 10110101, 11110011, 10110111, 11100111, 10110011

Then their values after the insertion of an 'A' would have been:

*1000000**1**, 10100100, 1011010**0**, 1011010**0**, 1111001**0**, 1011011**0**, 1110011**0**, 10110011*

(Again, the values in **bold** are the ones that were modified by the transformation)

From these examples we can infer that 1-LSB insertion usually has a 50% chance to change a LSB every 8 bits, thus adding very little noise to the original picture.

For 24-bit images the modification can be extended sometimes to the second or even the third LSBs without being visible. 8-bit images instead have a much more limited space where to choose colors, so it's usually possible to change only the LSBs without the modification being detectable.

B. Data Rate

The most basic of LSBs insertion for 24-bit pictures inserts 3 bits/pixel. Since every pixel is 24 bits, we can hide

3 hidden_bits/pixel / 24 data_bits/pixel = 1/8 hidden_bits/data_bits

So for this case we hide 1 bit of the embedded message for every 8 bits of the cover image.

If we pushed the insertion to include the second LSBs, the formula would change to:

6 hidden_bits/pixel / 24 data_bits/pixel = 2/8 hidden_bits/data_bits

And we would hide 2 bits of the embedded message for every 8 bits of the cover image. Adding a third-bit insertion, we would get:

9 hidden_bits/pixel / 24 data_bits/pixel = 3/8 hidden_bits/data_bits

Acquiring a data rate of 3 embedded bits every 8 bits of the image.

The data rate for insertion in 8-bit images is analogous to the 1 LSB insertion in 24-bit images, or 1 embedded bit every 8 cover bits.

We can see the problem in another light, and ask how many cover bytes are needed to send an embedded byte.

For 1-LSB insertion in 24-bit images or in 8-bit images this value would be 8/1*8 = 8 Bytes, for 2-LSBs insertion in 24-bit pictures it would be 8/2*8 = 4 Bytes, for 3-LSBs insertion it would be 8/3*8 = 21.33 Bytes.

C. Robustness

LSB insertion is very vulnerable to a lot of transformations, even the most harmless and usual ones.

Lossy compression, e.g. JPEG, is very likely to destroy it completely. The problem is that the "holes" in the Human Visual System that LSB insertion tries to exploit - little sensitivity to added noise - are the same that lossy compression algorithms rely on to be able to reduce the data rate of images.

Geometrical transformations, moving the pixels around and especially displacing them from the original grid, are likely to destroy the embedded message, and the only one that could allow recovery is a simple translation.

Any other kind of picture transformation, like blurring or other effects, usually will destroy the hidden data.

All in all, LSB insertion is a very little robust technique for data hiding.

D. Ease of detection/extraction

There is no theoretical outstanding mark of LSB insertion, if not a little increase of background noise.

It's very easy, instead, to extract LSBs even with simple programs, and to check them later to find if they mean something or not.

IV. ROBUSTNESS AND CRYPTOGRAPHY

Steganography tools aim to ensure robustness against modern forensic methods, such as statistical steganalysis. Such robustness may be achieved by a balanced mix of:

- a stream-based cryptography process;
- a data whitening process;
- an encoding process.

If the data is detected, cryptography also helps to minimize the resulting damage, since the data is not exposed, only the fact that a secret was transmitted. The sender may be forced to decrypt the data once it is discovered, but deniable encryption can be leveraged to make the decrypted data appear benign.

Strong steganography software relies on a multi-layered architecture with a deep, documented obfuscation process.

V. TOOLS COMPARISON

Program	Image files	Audio files	Video files	Document files	Other support
Anubis	BMP, JPG	?[clarification needed]	?[clarification needed]	?[clarification needed]	*Data being appended to the end of file*
BMPSecrets	BMP, JPG, TIFF, GIF	-	-	-	-
DarkCryptTC	BMP, JPG, TIFF, PNG, PSD, TGA, MNG	WAV	-	TXT, HTML, XML, ODT	*EXE, DLL, NTFS streams*
DeepSound	BMP	Audio CD, APE tag, FLAC, MP3, WAV, WMA	-	-	-
ImageSpyer G2	BMP, TIFF	-	-	-	-
MP3Stego	-	MP3	-	-	-
Mr. Crypto	BMP, PNG, TIFF	-	-	-	-
OpenPuff	BMP, JPEG, PNG, TGA	MP3, WAV	3gp, MP4, MPEG-1, MPEG-2, VOB, SWF, FLV	Pdf	-
OpenStego	BMP, PNG	-	-	-	-
OutGuess	JPEG, PNM	-	-	-	-
Outguess-rebirth	JPEG, PNM	-	-	-	-
PHP-Class StreamSteganogra	PNG	-	-	-	-

phy					
QuickStego / QuickCrypto	BMP, JPEG, GIF	-	-	-	-
Red JPEG	JPEG	-	-	-	-
S-Tools	BMP, GIF	Wav	-	-	*Unused floppy disk space*
Steg	BMP, PNG, JPEG, GIF	-	-	-	-
StegaMail	BMP, PNG	-	-	-	-
Steganographic Laboratory (VSL)	BMP, PNG, JPG, TIFF	-	-	-	-
Steganography Studio	BMP, PNG, GIF	-	-	-	-
SteganPEG	JPEG	-	-	-	-
StegFS	-	-	-	-	*Steganographic file system for Linux*
Steghide	JPG, BMP	WAV, AU	-	-	-
StegoShare	BMP, JPEG, PNG, GIF, TIFF	-	-	-	-

Source of Internet

VI. CONCLUSION

References

[1]. Habibi, M., Karimi, R., & Nosrati, M. (2013). Using SFLA andLSB for Text Message Steganography in 24-Bit RGBColor Imag- es. International Journal of Engineering, 2(3), 68-75.

[2]. Kessler, G. C. (2004). An overview of steganography for the computer forensics examiner. Forensic Science Communications, 6, 1-27.

[3]. Karim, M. (2011). A new approach for LSB based image steganog- raphy using secret key. International Conference on Computer and Information Technology on, 2011. 286-291.

[4]. Lee, Y.K., Bell, G., Huang, S.Y., Wang, R.Z. and Shyu, S.J. (2009).An advance Least-Significant-Bit Embedding Scheme for Ste- ganographic Encoding .Advance in Image and Video Technology. Berlin/Heidelberg:Springer,349-360.

[5]. Avcibas, I., Memon, N. and Sankur, B. (2003). Steganalysis using im- age quality metrics. Image Processing, IEEE Transactions on, 12,221-229.

[6]. Ahmed.A.M.(2012). A 2-Tire Datahiding technique using an im- proved exploiting Modification Direction Method and Huffman coding. Msc, Universiti Technologi Malaysia,Johor.

[7]. Bennett, K. (2004). Linguistic steganography: Survey, analysis, and robustness concerns for hiding information in text.

[8]. Chan, C. K. and Cheng, L. M. (2004). Hiding data in images by simple

[9]. LSB substitution. Pattern recognition, 37(3), 469-474.

[10]. Cheddad, A., Condell, J. ,Curran, K. and McKevitt, P. (2010). Digital image steganography: Survey and analysis of current methods. Sig- nal Processing, 90, 727-752.

[11]. Rabah, K. (2004). Steganography-the art of hiding data. Information Technology Journal, 3, 245-269.

[12]. Reddy, V. L., Subramanyam, A. and Reddy, P. C. (2011). Implemen- tation of LSB Steganography and its Evaluation for Various File Formats. Int. J. Advanced Networking and Applications, 2, 868-872.

[13]. Westfeld, A. and Wolf, G. (1998). Steganography in a video confer- encing system. Information Hiding, Springer, 32-47.

[14]. Westfeld, A. and Pfitzmann, A. (2000). Attacks on steganographic systems. Information Hiding, Springer, 61-76.

Software Testing Methodology for Finding Error and its Application

Komal Alwani
Research Scholar, Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Dr Pushpneel Verma
Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Abstract— **Testing is a process of finding errors while executing a program so that we get a zero defect software. It is aimed at evaluating the capability or usability of a program. Software testing is an important means of accessing quality of software. Though a lot of advancements have been done in formal methods and verification techniques, still we need software to be fully tested before it could be handled to the customer side. Thus there are a number of testing techniques and tools made to accomplish the task. Software testing in very important activity in software development. It is one of the important activity which require lot of time and labour. Different testing techniques are used to find bugs in the software. Testing is involved at different stages of software development like unit testing, integration testing, system testing, acceptance testing etc. Different testing techniques namely dynamic testing, functional testing and structural testing are used to test software. Software**

Keywords: Stage; Level; Data; SoftwareTesting,Time.

I. INTRODUCTION

There is scope for automation in the activities of testing but testers experience is very much important for successful testing. Software testing is component of software quality control. Different types of tests are used to testing like unit testing, integration testing, acceptance testing, system testing are used to test a system. Techniques of testing like static testing, functional testing, dynamic testing and structural testing have been used to test the system. We discussed a case study on application of SBST techniques for an industrial embedded software. During the experiments we were able to increase the total branch coverage of existing test cases. So we can say that the employed SBST techniques indicate beneficial results. However, we observed that randomly generated inputs were as effective as more informed search algorithms in many (but not all) cases, which indicate that there are many branches that are easy to cover.

Various difficulties were encountered at the implementation phase due to the specialized execution environment of the embedded software. Therefore, our prototype implementation was highly adapted to the system under test, instead of following a more generic structure. Another important characteristic of the system under test is its complexity and the specialized domain.

There is an increasing integration of software into many products. This makes software quality a relevant factor in the overall quality of all such products. However, systems engineers and integrators are not software engineering experts and, in particular, have little or no software testing experience. One option to ensure proper quality of the software being integrated is to involve dedicated software engineers to handle software development and testing.

The are some software testing includes. [6][7][8]

- The better it works the more efficiently it can be tested.
- Better the software can be controlled more the testing can be automated and optimized.
- The fewer the changes, the fewer the disruption to testing.
- A successful test is the one that uncovers an undiscovered error.
- Testing is a process to identify the correctness and completeness of the software.
- The general objective of software testing is to affirm the quality of software system by systematically exercising the software in carefully controlled circumstances.

Classified by purpose software testing can be divided into [4]

1. Correctness Testing
2. Performance Testing
3. Reliability Testing
4. Security Testing

Sometimes the cost of fixing an error may affect a decision not to fix an error. This is particularly true if the error is found late in the lifecycle. For example, when an error has caused a failure during system test and the location of the error is found to be in the requirements or design, correcting that error can be expensive. Sometimes the error is allowed to remain and the fix deferred until the next version of the software. Persons responsible for these decisions may justify them simply on the basis of cost or on an analysis which shows that the error, even when exposed, will not cause a critical failure. Decision makers must have confidence in the analyses used to identify the impact of the error, especially for software used in high integrity systems.

A strategy for avoiding the high costs of fixing errors late in the lifecycle is to prevent the situation from occurring altogether, by detecting and correcting errors as early as possible. Studies have shown that it is much more expensive to correct software requirements deficiencies late in the development effort than it is to have correct requirements from the beginning [STSC].

II. IMPLICATIONS

Performance Testing' involve all the phases as the mainstream testing life cycle as an independent discipline which involve strategy such as plan, design, execution, analysis and reporting. This testing is conducted to evaluate the compliance of a system or component with specified performance requirement. [2] Evaluation of a performance of any software system includes resource usage, throughput and stimulus response time. By performance testing we can measure the characteristics of performance of any applications. One of the most important objectives of performance testing is to maintain a low latency of a website, high throughput and low utilization. [5]

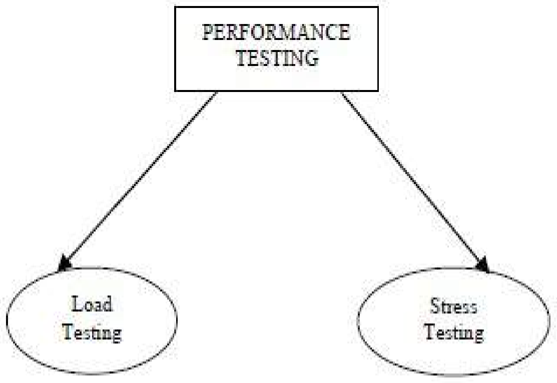


Fig. 1. Represent two types of performance testing

Some of the main goals of performance testing are: [5]

- ➢ Measuring response time of end to end transactions.
- ➢ Measurement of the delay of network between client and server.
- ➢ Monitoring of system resources which are under various loads.

Some of the common mistakes which happen during performance testing are: [5]

- ➢ Ignoring of errors in input.
- ➢ Analysis is too complex.
- ➢ Erroneous analysis.
- ➢ Level of details is inappropriate.
- ➢ Ignore significant factors.
- ➢ Incorrect Performance matrix.
- ➢ Important parameter is overlooked.
- ➢ Approach is not systematic.

There are seven different phases in performance testing process: [5]

- ➢ Phase 1 – Requirement Study
- ➢ Phase 2 – Test plan
- ➢ Phase 3 – Test Design
- ➢ Phase 4 – Scripting
- ➢ Phase 5 – Test Execution
- ➢ Phase 6 – Test Analysis
- ➢ Phase 7 – Preparation of Report

III. TESTING PROCESS

*Traditional waterfall development model**[edit]***

A common practice in waterfall development is that testing is performed by an independent group of testers. This can happen:

- after the functionality is developed, but before it is shipped to the customer.[64] This practice often results in the testing phase being used as a project buffer to compensate for project delays, thereby compromising the time devoted to testing.[13]:145–146
- at the same moment the development project starts, as a continuous process until the project finishes.[65]

However, even in the waterfall development model, unit testing is often done by the software development team even when further testing is done by a separate team.[66]

Further information: Capability Maturity Model Integration and Waterfall model

*1) Agile or XP development model**[edit]***

In contrast, some emerging software disciplines such as extreme programming and the agile software development movement, adhere to a "test-driven software development" model. In this process, unit tests are written first, by the software engineers (often with pair programming in the extreme programming methodology). The tests are expected to fail initially. Each failing test is followed by writing just enough code to make it pass.[67] This means the test suites are continuously updated as new failure conditions and corner cases are discovered, and they are integrated with any regression tests that are developed. Unit tests are maintained along with the rest of the software source code and generally integrated into the build process (with inherently interactive tests being relegated to a partially manual build acceptance process).

The ultimate goals of this test process are to support continuous integration and to reduce defect rates.[68][67]

This methodology increases the testing effort done by development, before reaching any formal testing team. In some other development models, most of the test execution occurs after the requirements have been defined and the coding process has been completed.

*2) A sample testing cycle**[edit]***

Although variations exist between organizations, there is a typical cycle for testing.[2] The sample below is common among organizations employing the Waterfall development model. The same practices are commonly found in other development models, but might not be as clear or explicit.

- Requirements analysis: Testing should begin in the requirements phase of the software development life cycle. During the design phase, testers work to determine what aspects of a design are testable and with what parameters those tests work.

- Test planning: Test strategy, test plan, testbed creation. Since many activities will be carried out during testing, a plan is needed.
- Test development: Test procedures, test scenarios, test cases, test datasets, test scripts to use in testing software.
- Test execution: Testers execute the software based on the plans and test documents then report any errors found to the development team. This part could be complex when running tests with a lack of programming knowledge.
- Test reporting: Once testing is completed, testers generate metrics and make final reports on their test effort and whether or not the software tested is ready for release.
- Test result analysis: Or Defect Analysis, is done by the development team usually along with the client, in order to decide what defects should be assigned, fixed, rejected (i.e. found software working properly) or deferred to be dealt with later.
- Defect Retesting: Once a defect has been dealt with by the development team, it is retested by the testing team.
- Regression testing: It is common to have a small test program built of a subset of tests, for each integration of new, modified, or fixed software, in order to ensure that the latest delivery has not ruined anything and that the software product as a whole is still working correctly.
- Test Closure: Once the test meets the exit criteria, the activities such as capturing the key outputs, lessons learned, results, logs, documents related to the project are archived and used as a reference for future projects.

IV. SOFTWARE APPLICATION TESTING

Application Testing is defined as a software testing type, conducted through scripts with the motive of finding errors in software. It deals with tests for the entire application. It helps to enhance the quality of your applications while reducing costs, maximizing ROI, and saving development time.

Application Testing is categorized into two segments.

- Web Application testing
- Desktop Application testing

Application Testing	**Types of Testing**
• Web Application Testing	• Functional and Performance Testing • Cross-browser Testing • Load and stress Testing • Regression and Compliance Testing • User Acceptance Testing • Beta Testing • Exploratory and Smoke Testing • Multilanguage support and compatibility Testing
• Desktop Application Testing	• UI Testing • Usability Testing • Performance Testing • Compatibility Testing (Software/ Hardware) • Functional Testing • Security Testing
• Mobile Application Testing	• UI Testing • Rule based Testing • Regression Testing • Functional Testing • Security Testing

V. APPLICATION TESTING METHODOLOGIES

Testing methodologies is a different way of ensuring that a software application is fully tested. Unorganized and poor testing methodology can lead to an unstable product.

There are three ways Testing is carried out.

- Black Box Testing

- White Box Testing
- Grey Box Testing

VI. SOFTWARE ERROR TESTING

Testing is the process of identifying defects, where a defect is any variance between actual and expected results. “A mistake in coding is called Error, error found by tester is called Defect, defect accepted by development team then it is called Bug, build does not meet the requirements then it Is Failure.”
Defect can be categorized into the following:

Wrong: When requirements are implemented not in the right way. This defect is a variance from the given specification. It is Wrong!

Missing: A requirement of the customer that was not fulfilled. This is a variance from the specifications, an indication that a specification was not implemented, or a requirement of the customer was not noted correctly.

Extra: A requirement incorporated into the product that was not given by the end customer. This is always a variance from the specification, but may be an attribute desired by the user of the product. However, it is considered a defect because it’s a variance from the existing requirements.

ERROR: An error is a mistake, misconception, or misunderstanding on the part of a software developer. In the category of developer we include software engineers, programmers, analysts, and testers. For example, a developer may misunderstand a de-sign notation, or a programmer might type a variable name incorrectly – leads to an Error. It is the one which is generated because of wrong login, loop or due to syntax. Error normally arises in software; it leads to change the functionality of the program.

BUG: A bug is the result of a coding error. An Error found in the development environment before the product is shipped to the customer. A programming error that causes a program to work poorly, produce incorrect results or crash. An error in software or hardware that causes a program to malfunction. Bug is terminology of Tester.

FAILURE: A failure is the inability of a software system or component to perform its required functions within specified performance requirements. When a defect reaches the end customer it is called a Failure. During development Failures are usually observed by testers.

FAULT: An incorrect step, process or data definition in a computer program which causes the program to perform in an unintended or unanticipated manner. A fault is introduced into the software as the result of an error. It is an anomaly in the software that may cause it to behave incorrectly, and not according to its specification. It is the result of the error.

References

[1]. Harrold Mary Jean, and Gregg Rothermel. "Performing data flow testing on classes." ACM SIGSOFT Software Engineering Notes. Vol. 19. No. 5. ACM, 1994.[acm.pdf]

[2]. Claessen Koen, and John Hughes. "QuickCheck: a lightweight tool for random testing of Haskell programs." Acm sigplan notices 46.4 (2011): 53-64.

[3]. Vilkomir Sergiy A., Kalpesh Kapoor, and Jonathan P. Bowen. "Tolerance of control-flow testing criteria." Computer Software and Applications Conference, 2003. COMPSAC 2003. Proceedings. 27th Annual International. IEEE, 2003.[control ieee]

[4]. A. P. Mathur, “Foundation of Software Testing”, Pearson/Addison Wesley, 2008.

[5]. IEEE Standard 829-1998, “IEEE Standard for Software Test Documentation” pp.1-52, IEEE Computer Society, 1998.

[6]. D. Gelperin and B. Hetzel, “The Growth of Software Testing”, Communications of the ACM, Volume 31 Issue 6, June 1988, pp. 687- 695[history of st]

[7]. D. Richardson, O. O’Malley and C. Tittle, “Approaches to specification-based testing”, ACM SIGSOFT Software Engineering Notes, Volume 14 , Issue 9, 1989, pp. 86 – 96[Approaches to specification-based testing]

[8]. S. Rapps and E. J. Weyuker, “Selecting Software Test Data Using Data Flow Information,” IEEE Transactions on Software Engineering¸ April 1985, pp. 367-375

A Study on Next Generation Cloud Computing

Chandan Rankawat
Research Scholar, Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Er. Harish Chandra Maurya
Asstt. Prof., Department of CSE
Bhagwant University
Ajmer, Rajasthan, India

Abstract—**It reckons in of a compilation of integrated and networked hardware, software and internet infrastructure. It has various avails atop grid computing and other computing Like real clouds which are the collection of water molecules, the term _cloud‘ in cloud computing is the collection of networks. The user can use the modalities of cloud computing boundlessly whenever demanded. Instead of setting up their own physical infrastructure, the users ordinarily prefer a mediator provider for the service of the internet in cloud computing.**

Keywords: Stage; Image; Type; Components,etc.

I. INTRODUCTION

Every entity that is being part of a system is having a definite evolution, As far as Cloud Computing is concerned, there is no exact date which mentions the evolution of it, However in 1960s, John McCarthy, Douglas Parkhill, and others explored the idea of computing as a public utility, because of the existence of mainframe computers, during that period, the clients were accessing the central computing power through dummy terminals, which enable the clients to access the mainframe computer. With high cost and maintenance, it was not feasible for the organizations to buy these critical resources, and was the most challenging task for the big companies and organization to stay in the business market, and then there arose the concept of shared access to the single computing system in order to save the cost of buying separate machines. Evolution in Information Technology is not all of a sudden process rather it is a step-by-step transformation that brings a lot to cherish for organizations, companies. IMB launch the operating system in 1970 known as Virtual Machine (VM), this enabled the companies and organizations to run their operations on the operating systems simultaneously on more than one system with own memory and processing unit, VM became the initial phase towards the evolution of new technology known as Virtualization, collective collaboration of different computing platforms like Centralized, Parallel, Cluster, Distributed and Grid Computing gave birth of today’s most talked computing paradigm known as Cloud Computing.

A development environment or platform is given to the consumers as a service in PaaS, upon which user can deploy their own software and coding. The customer has the liberty to construct his own applications that can run on the provider's infrastructure [5]. Product as a service providers offers a predefined composition of operating system and application server to obtain the management capacity of the applications. For example, LAMP (Linux, Apache, MySQL, and PHP), J2EE, Ruby etc.

Strategies implemented to mitigate failures on the cloud include using redundant compute systems in a data center, multiple zones and back up data centers in individual zones. However, alternate models of using cloud infrastructure instead of using data centers from a single provider have been proposed in recent years [2]. In this paper, we consider the multi-cloud, microcloud and cloudlet, ad hoc cloud and heterogeneous cloud to demonstrate the trends in changing infrastructure of the cloud. The feasibility of these have been reported in literature and will find real deployment of workloads in next generation cloud computing. Fig. 2 shows the different layers of the cloud stack where changes need to be accommodated due to the evolving infrastructure. We consider nine layers of abstraction that contribute to the cloud stack, namely network (bottom of the stack), storage, servers, virtualisation, operating system, middleware, runtime, data and application (top of the stack). For facilitating multi-cloud environments and ad hoc clouds, changes will be required from the middleware layer and upwards in the stack. Heterogeneous clouds can be achieved with changes two further layers down the stack from the virtualisation layer. Microclouds and cloudlet infrastructure may require redesign of the servers that are employed and therefore changes are anticipated from the server layer.

II. TYPES OF CLOUD COMPUTING

Public Cloud: The public cloud is a computing service supplied by the third party providers atop the public internet [6]. These services are available for any user who wants to use them and they have to pay only for the services they consumed.

Private Cloud: The computing services provided over the internet or private network come under the private cloud and these services are offered only to the selected users in place of common people [1,6]. A higher security and privacy is delegated by private clouds through the firewall and internal hosting .

Hybrid Cloud: Hybrid cloud is the combination of public cloud and private cloud. In the hybrid cloud, each cloud can be managed independently but data and applications can be shared among the clouds in the hybrid cloud [1, 6].

III. COMPONENTS OF CLOUD COMPUTING

Cloud computing has three basic components as follows
Client Computers: The end user can interact with the cloud using the client computers.
Distributed Servers: The servers are distributed among the different places but acts like they as working with each other.
Data Centres: Data centres are the compilation of servers.

IV. NEXT GENERATION CLOUD COMPUTING

Multi-cloud: The use of multi-clouds are increasing, but there are hurdles that will need to be overcome. For example, common APIs to facilitate multi-cloud need to account for different types of resources offered by multiple providers. This is not an easy given that more resources are rapidly added to the cloud marketplace and there are no unified catalogues that report a complete set of resources available on the cloud. Further, the abstractions, including network and storage architectures differ across providers, which makes the adoption of multi-cloud bespoke to each application rather than using a generic platform or service. Along with the different resources, hypervisors, and software suites employed, the pricing and billing models are significantly different across providers, all of which results in significant programming effort required for developing a multi-cloud application.
Microcloud and cloudlet: f electricity to provide a centralised computing infrastructure. This is a less sustainable trend, and alternate low power and low cost solutions are proposed. There are recent efforts to decentralise computing towards the edge of the network for making computing possible closer to where user data is generated [12]. Small sized, low cost and low power computing processors co-located with routers and switches or located in dedicated spaces closer to user devices, referred to as microclouds are now developed for this purpose [13–15]. However, there are no public deployments given the challenges in networking microcloud installations over multiple sites. In the UK, there are efforts to connect microclouds for general purpose computing.8 Microclouds lend themselves in reducing latency of applications and minimising the frequency of communication between user devices and data centers. Odroid boards9 and Raspberry Pis10 for example are used to develop microclouds. However, integration of microclouds to the existing computing ecosystem is challenging and efforts are being made in this direction [16].
Ad hoc cloud: The concept of ad hoc clouds is based on the premise of ad hoc computing in that underutilised resources, such as servers owned by organisations can be harnessed to create an elastic infrastructure [2,1]. This is in contrast to existing cloud infrastructure which is largely data center based and in which the resources available are known beforehand. However, the context of an ad hoc cloud is changing with increasing connectivity of a large variety of resources to the cloud [2]. This is becoming popular for smaller mobile devices, such as smartphones [3,4], which on an average have less than a 25% per hour of usage [5,6]. The spare resources of smartphones can contribute to creating an ad hoc infrastructure (such as cloudlets) that supports low latency computing for non-critical applications in public spaces and transportation systems. The assumption here is that one device is surrounded by a large number of devices that will complement computing for the former device. Although such an infrastructure is not reliable, it may be used in conjunction with existing data centers to enhance connectivity.
Heterogeneous cloud: The second is related to low-level heterogeneity at the infrastructure level, in which different types of processors are combined to offer VMs with heterogeneous compute resources. In this paper, the latter is referred to as heterogeneous clouds. While supercomputers have become more heterogeneous by employing accelerators, such as NVIDIA GPUs or Intel Xeon Phis, cloud data centers mostly employ homogeneous architectures [9]. More recently heterogeneous cloud data center architectures have been proposed.14 In the vendor arena, Amazon along with other providers offer GPU-based VMs, but accelerators are not yet fully integrated into the computing ecosystem. This is because it is not yet possible for a programmer to fully develop and execute code oblivious to the underlying hardware. There are a number of efforts in this direction, but the key challenge is achieving a high-level abstraction that can be employed across multiple architectures, such as GPUs, FPGAs and Phis [3]. Further applications that already execute on the cloud cannot be scheduled onto heterogeneous resources.

V. BIG DATA COMPUTING

Firstly, data processing and resource management on distributed cloud nodes29 [5]. Whether they be ad hoc clouds, heterogeneous clouds or distributed clouds, there needs to be platforms that can take into account the ad hoc nature of nodes that may process data in a distributed cloud setting, the heterogeneity of processors and platforms that scale from low power processors to high-end processors without significant programming efforts. Secondly, building models for analytics that scale both horizontally and vertically. Current models typically scale horizontally across multiple nodes in a data center or across nodes in multiple data centers. In the future, models that scale vertically from low end processors to data center nodes will need to be developed. Thirdly, software stacks for end-to-end processing [6]. This relates to both the first and second challenges. Currently, most big data solutions assume a centralised cloud as the compute resource, but integrating microclouds, cloudlets or traffic routing edge nodes in the software stack will need to be addressed. In the real-world, the volume of unstructured data as opposed to structured data is increasing. Often unstructured data is interconnected (for example, generated from social networks) and takes the form of natural language. One key challenge is achieving accurate and actionable knowledge from unstructured data. To address this challenge, one approach will be to transform unstructured data to structured networks and then to knowledge, referred to as data-to-network-to-knowledge (D2N2K).

VI. CONCLUSION

Cloud computing describes both a platform and a type of application. The privilege given to cloud applications is that they can extended to be accessible through the Internet. These cloud applications use large data centers and powerful servers that host Web applications and Web services. Both the changing cloud infrastructure and emerging computing architecture will impact a number of areas. They will play a vital role in improving connectivity between people and devices to facilitate the Internet-of-Things paradigm. The area of data intensive computing will find novel techniques to address challenges related to dealing with volume of data. New services, such as containers, acceleration and function, is anticipated become popular. A number of research areas will find convergence with next generation cloud systems to deliver self-learning systems. Cloud computing services can also grow and shrink according to need. Cloud computing is particularly valuable to small and medium businesses, where effective and affordable IT tools are critical to helping them become more productive without spending lots of money on in-house resources and technical equipment.

References

[1]. R. Buyya, C.S. Yeo, S. Venugopal, J. Broberg, I. Brandic, Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility, Future Gener. Comput. Syst. 25 (6) (2009) 599–616.

[2]. N. Grozev, R. Buyya, Inter-cloud architectures and application brokering: taxonomy and survey, Softw. - Pract. Exp. 44 (3) (2014) 369–390.

[3]. Y. Mansouri, A.N. Toosi, R. Buyya, Data storage management in cloud environments: Taxonomy, survey, and future directions, ACM Comput. Surv. (2017) 1–51.

[4]. D. Petcu, G. Macariu, S. Panica, C. Crciun, Portable cloud applications: From theory to practice, Future Gener. Comput. Syst. 29 (6) (2013) 1417–1430.

[5]. Z. Wu, H.V. Madhyastha, Understanding the latency benefits of multi-cloud webservice deployments, SIGCOMM Comput. Commun. Rev. 43 (2) (2013) 13–20.

[6]. D. Bernstein, E. Ludvigson, K. Sankar, S. Diamond, M. Morrow, Blueprint for the intercloud - protocols and formats for cloud computing interoperability, in: Proceedings of the 4th International Conference on Internet and Web Applications and Services, 2009, pp. 328–336.

[7]. K. Zhang, X. Zhou, Y. Chen, X. Wang, Y. Ruan, Sedic: Privacy-aware data intensive computing on hybrid clouds, in: Proceedings of the 18th ACM Conference on Computer and Communications Security, 2011, pp. 515–526.

[8]. X. Xu, X. Zhao, A framework for privacy-aware computing on hybrid clouds with mixed-sensitivity Data, in: 17th IEEE International Conference on High Performance Computing and Communications, 7th IEEE International Symposium on Cyberspace Safety and Security, and 12th IEEE International Conference on Embedded Software and Systems, 2015, pp. 1344–1349.

[9]. A.N. Toosi, R.O. Sinnott, R. Buyya, Resource provisioning for data-intensive applications with deadline constraints on hybrid clouds using aneka, Future Gener. Comput. Syst. (2017). http://dx.doi.org/10.1016/j.future.2017.05.

[10]. B. Rochwerger, C. Vzquez, D. Breitgand, D. Hadas, M. Villari, P. Massonet, E. Levy, A. Galis, I.M. Llorente, R.S. Montero, Y. Wolfsthal, K. Nagin, L. Larsson, F. Galn, An architecture for federated cloud computing, in: Cloud Computing, John Wiley & Sons, Inc., 2011, pp. 391–411.

[11]. R. Buyya, R. Ranjan, R.N. Calheiros, InterCloud: Utility-oriented federation of cloud computing environments for scaling of application services, in: Proceedings of the 10th International Conference on Algorithms and Architectures for Parallel Processing, 2010, pp. 13–31.

[12]. B. Varghese, N. Wang, S. Barbhuiya, P. Kilpatrick, D.S. Nikolopoulos, Challenges and opportunities in edge computing, in: IEEE International Conference on Smart Cloud, 2016, pp. 20–26.

[13]. F.P. Tso, D.R. White, S. Jouet, J. Singer, D.P. Pezaros, The glasgow raspberry Pi cloud: A scale model for cloud computing infrastructures, in: 33rd IEEE International Conference on Distributed Computing Systems Workshops, 2013, pp. 108–112.

[14]. A. Sathiaseelan, A. Lertsinsrubtavee, A. Jagan, P. Baskaran, J. Crowcroft, Cloudrone: Micro clouds in the sky, in: Proceedings of the 2nd Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use, 2016, pp. 41–44.

[15]. Y.S.S.A. Elkhatib, B.F. Porter, H.B. Ribeiro, M.F. Zhani, J. Qadir, E. Rivire, On using micro-clouds to deliver the fog, IEEE Internet Comput. 21 (2) (2016) 8–15.

[16]. M. Villari, M. Fazio, S. Dustdar, O. Rana, R. Ranjan, Osmotic computing: A new paradigm for edge/cloud integration, IEEE Cloud Comput. 3 (6) (2016) 76–83.

A Study on Neural Network approaches in the Eye Detection

Nikita Bhardwaj
Research Scholar, Department of ECE
Bhagwant University
Ajmer, Rajasthan, India

Er. Mithlesh Kumar Singh
Department of ECE
Bhagwant University
Ajmer, Rajasthan, India

Abstract— **In this paper we propose a method for Eye Detection and Neural Networks. A lot of research work has been published in the field of eye detection so far. Various techniques have been proposed using template matching, IR based approaches, feature based approach, Skin detection method, Hough transform method, Eigen space method or combination of these for eye detection. The skin region helps determining the approximate eye position and eliminates a large number of false eye candidates. There are various color spaces available to detect skin region such as HSV, RGB, YCbCr, and NTSC spaces to increase the efficiency of eye detection. Furthermore, it is computationally expensive and requires good image contrast. Since facial features exhibit large variability among subjects and under different experimental conditions, feature-based approaches tend to work well with some subjects in some experiments but can fail completely with other subjects under the same experimental conditions. Also, many feature-based techniques are limited to relatively frontal view of faces and therefore a more flexible and robust approach to eye detection is needed in order to cope with the expected variability of eye features for the full range of subjects' head movements.**

Keywords: Neural; Network; Method; Components,etc.

I. INTRODUCTION

The skin region helps determining the approximate eye position and eliminates a large number of false eye candidates. There are various color spaces available to detect skin region such as HSV, RGB, YCbCr, and NTSC spaces to increase the efficiency of eye detection. The data was obtained for all the color space components, and was fitted to a Gaussian curve and peak value determines the color of skin. Pentland et al. [7] proposed an Eigen space method for eye and face detection. If the training database is variable with respect to appearance, orientation, and illumination, then this method provides better performance than simple template matching. But the performance of this method is closely related to the training set used and this method also requires normalized sets of training and test images with respect to size and orientation. Another popular eye detection method is obtained by using the Hough transform. This method is based on the shape feature of an iris and is often used for binary valley or edge maps. Hough transform is a general technique for identifying the locations and orientations of certain types of features in a digital image and used to isolate features of a particular shape within an image. Lam and Yuen [6] noted that the Hough transform is robust to noise, and can resist to a certain degree if occlusion and boundary effects. Akihiko Torii and Atsushi Imiya [7] proposed a randomized Hough transform based method for the detection of great circles on a sphere. Cheng Z. and Lin Y [8] proposed a new efficient method to detect ellipses in gray-scale images, called Restricted Randomized Hough transform. Various other methods that have been adopted for eye detection include wavelets, principal component analysis, fuzzy logic, evolutionary computation and hidden markov models. Huang and Wechsler [10] perform the task of eye detection by using optimal wavelet packets for eye representation and radial basis functions for subsequent classification of facial areas into eye and non-eye regions. Filters based on Gabor wavelets to detect eyes in gray level images are used in [11]. Talmi et al. and Pentland et al. [12,13] use principal component analysis to describe and represent the general characteristics of human eyes with only very few dimensions. In[7] Eigeneyes are calculated by applying Karhunen-Loeve-Transformation to represent the major characteristics of human eyes and are stored as reference patterns for the localization of human eyes in video images.

II. APPROACHES FOR EYE DETECTION

Pattern-based approaches for eye detection are concerned with the automatic discovery of regularities in data through the use of computer algorithms and with the use of these regularities to detect the eyes. In [15]-[17], Eigen-Eye is developed to detect eye regions. Eigen-Eye uses Principal Component Analysis (PCA) to compute a set of basis images that provides a low dimensional representation of all possible eye images. To classify an image pattern as an eye or a non-eye, the image pattern is mapped to the space formed by the basis images and a similarity measure is used Convolutional Neural Networks for Eye Detection in Remote Gaze Estimation Systems Jerry, Chi Ling Lam and Moshe Eizenman to classify this pattern as an eye or a non-eye. The Eigen-Eye approach can fail to detect eye images that cannot be described by a simple linear combination of the basis images. In [18] and [19], a large database of eye and non-eye images is used to train a Support Vector Machine (SVM) to detect eyes. One of the disadvantages of SVM is that the model designer has to make subjective decisions about the type of kernel to be used before the training procedure. This might not be ideal because it is difficult to justify quantitatively the choice of a specific kernel for a specific application. In [20] and [21], Multi-Layer Feed forward Neural Networks are used to detect eye regions. Unlike Eigen-Eye and SVM, Neural Networks learn discriminative features about the eye from the training data. Since Neural Networks provide the best detection performance when the input data set exhibits large variability [22], a specific configuration of Neural Networks, Convolutional Neural Networks (CNN), was developed and optimized for the detection of eye regions in video images from the eye-tracker's camera. Secondly, CNN emphasizes a key property of images which is that nearby pixels are much more likely to be correlated than more distant pixels. It achieves this property by extracting features that depend only on small sub-regions of the image. Information from such features is merged in later stages of processing in order to detect more complex features, and ultimately to yield information about the image as a whole. Lastly, in many applications that use Neural Networks [27]-[29], the original image is first preprocessed and the processed image is then fed into the Neural Networks for classification. This preprocessing step is essential, for example, for image intensity normalization. CNN does not require any preprocessing steps. It learns to build the preprocessing module and the classification module in a single integrated scheme.

III. NEURAL NETWORK

The network will receive the 960 real values as a 960-pixel input image (Image size ~ 32 x 20). It will then be required to identify the eye by responding with a output vector[25]. The output vectors represent a eye or non-eye. To operate correctly the network should respond with a 1 if eye is presented to the network else output vector should be 0 [25]. In addition, the network should be able to handle non-eye. In practice the network will not receive a perfect image of eye which represented by vector as input.

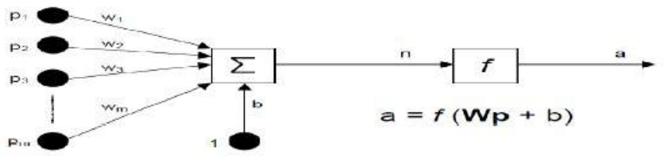


Figure 1.1: Neural Network Architecture (Source: Vijayalaxmi Paper)

Figure 1.2: Experimental results showing Original image & Eyes detected (Source: Vijayalaxmi Paper)

Figure 1.3: Images with failure in the location of the eyes (Source: Vijayalaxmi Paper)

IV. EYE CENTER POSITIONING

Although the 1st set of CNNs outputs the eye region and the eye class (left or right), it still lacks precise positioning of the eye center. The existing eye detectors usually treat the center of the eye region as the eye center, which is inaccurate if the subject is not looking straightwards. To locate the actual eye center, we built the 2nd set of CNNs that locates the pupil region in the eye region. The 2nd CNNs' architecture is shown in Figure 5. Compared to the 1st set of CNNs, the structure of the 2nd network is relatively simple. It is composed of a convolution layer, an average pooling layer, a fully connected layer, and a logistic perceptron. The input eye region for this set of CNNs depends on the first CNNs' output. The size of the other layers was adapted accordingly, and we chose the center of pupil region as the eye center.

V. THE POINT-OF-GAZE (POG)

Most modern remote gaze estimation systems are based on the analysis of specific eye features extracted from video images [10]. These specific eye features are the pupil center and a set of corneal reflections. The corneal reflections are virtual images of the light sources that illuminate the eyes. When the field of view of the eye tracker's video camera is limited, it is relatively easy to determine the eye features (Figure 1a). But, when the field of view of the camera is increased to allow for a larger range of head movements, the detection of eye features becomes much more difficult (Figure 1b). With a larger field of view, the number of false detections increases and the performance of the remote gaze estimation system deteriorate. It is therefore essential to develop algorithms that can detect reliably regions in the image that contain the eyes. Eye detection can be broadly classified into feature-based approaches and pattern-based approaches. Feature-based approaches explicitly model facial features to detect eye regions. In [11], dark regions are assumed to be associated with pupils and are searched within the face by iteratively thresholding the face image to locate eye candidates. Two eye candidates that satisfy certain anthropometric constraints are then identified as the detected eyes. This approach is reliable as long as subjects remain relatively frontal with respect to the camera and they don't wear eyeglasses. In [12]-[14], deformable eye templates are designed to fit the best representation of eyes in images. In this technique, eyes are detected through a recursive process that minimizes a specific energy function. In [12], the energy function is designed such that the energy is minimized when the total brightness inside the pupil candidate is small. In [18] and [19], a large database of eye and non-eye images is used to train a Support Vector Machine (SVM) to detect eyes. One of the disadvantages of SVM is that the model designer has to make subjective decisions about the type of kernel to be used before the training procedure. This might not be ideal because it is difficult to justify quantitatively the choice of a specific kernel for a specific application. In [20] and [21], Multi-Layer Feed forward Neural Networks are used to detect eye regions. Unlike Eigen-Eye and SVM, Neural Networks learn discriminative features about the eye from the training data. Since Neural Networks provide the best detection performance when the input data set exhibits large variability [22], a specific configuration of Neural Networks, Convolutional Neural Networks (CNN), was developed and optimized for the detection of eye regions in video images from the eye-tracker's camera.

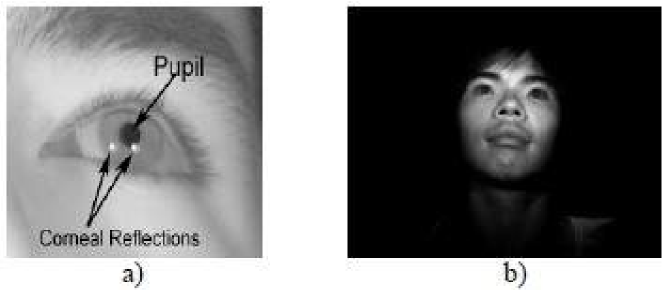


Figure 1.4: Eye Tracker's Images a) small field of view, b) larger field of view (Source: Jerry, Chi Ling Lam and Moshe Eizenman Paper)

References

[1]. S. Chen and C. Liu, "Eye detection using discriminatory Haar features and a new efficient SVM," Image and Vision Computing, vol. 33, pp. 68–77, 2015. View at Publisher · View at Google Scholar · View at Scopus

[2]. R. Sharma and A. Savakis, "Lean histogram of oriented gradients features for effective eye detection," Journal of Electronic Imaging, vol. 24, no. 6, Article ID 063007, 2015. View at Publisher · View at Google Scholar · View at Scopus

[3]. M. Leo, D. Cazzato, T. De Marco, and C. Distante, "Unsupervised approach for the accurate localization of the pupils in near-frontal facial images," Journal of Electronic Imaging, vol. 22, no. 3, Article ID 033033, 2013. View at Publisher · View at Google Scholar · View at Scopus

[4]. M. Leo, D. Cazzato, T. De Marco, and C. Distante, "Unsupervised eye pupil localization through differential geometry and local self-similarity matching," PLoS ONE, vol. 9, no. 8, Article ID e102829, 2014. View at Publisher · View at Google Scholar · View at Scopus

[5]. T. D'Orazio, M. Leo, and A. Distante, "Eye detection in face images for a driver vigilance system," in Proceedings of the 2004 IEEE Intelligent Vehicles Symposium, pp. 95–98, ita, June 2004. View at Scopus

[6]. C. Gou, Y. Wu, K. Wang, K. Wang, F.-Y. Wang, and Q. Ji, "A joint cascaded framework for simultaneous eye detection and eye state estimation," Pattern Recognition, vol. 67, pp. 23–31, 2017. View at Publisher ·View at Google Scholar · View at Scopus

[7]. H. Kim, J. Jo, K.-A. Toh, and J. Kim, "Eye detection in a facial image under pose variation based on multi-scale iris shape feature," Image and Vision Computing, vol. 57, pp. 147–164, 2017. View at Publisher· View at Google Scholar · View at Scopus

[8]. D. Xie, L. Zhang, and L. Bai, "Deep Learning in Visual Computing and Signal Processing," Applied Computational Intelligence and Soft Computing, vol. 2017, pp. 1–13, 2017. View at Publisher · View at Google Scholar

[9]. W. Chinsatit and T. Saitoh, "CNN-Based Pupil Center Detection for Wearable Gaze Estimation System," Applied Computational Intelligence and Soft Computing, vol. 2017, pp. 1–10, 2017. View at Publisher ·View at Google Scholar

[10]. W. Fuhl, PupilNet: convolutional neural networks for robust pupil detection, arXiv preprint arXiv1601.04902, convolutional neural networks for robust pupil detection, PupilNet, 2016.

[11]. B. Amos, B. Ludwiczuk, and M. Satyanarayanan, Openface: A general-purpose face recognition library with mobile applications, CMU School of Computer Science, Openface, 2016.

[12]. P. Viola and M. J. Jones, "Robust real-time face detection," International Journal of Computer Vision, vol. 57, no. 2, pp. 137–154, 2004. View at Publisher · View at Google Scholar · View at Scopus

[13]. J. Deng, W. Dong, and R. Socher, "ImageNet: a large-scale hierarchical image database," in Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 248–255, Miami, Fla, USA, June 2009. View at Publisher · View at Google Scholar

[14]. A. Villanueva, V. Ponz, L. Sesma-Sanchez, M. Ariz, S. Porta, and R. Cabeza, "Hybrid method based on topography for robust detection of iris center and eye corners," ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), vol. 9, no. 4, article 25, 2013. View at Publisher· View at Google Scholar · View at Scopus

[15]. O. Jesorsky, K. J. Kirchberg, and R. W. Frischholz, "Robust face detection using the hausdorff distance," in Audio- and Video-Based Biometric Person Authentication: Third International Conference, AVBPA 2001 Halmstad, Sweden, June 6–8, 2001 Proceedings, J. Bigun and F. Smeraldi, Eds., vol. 2091 of Lecture Notes in Computer Science, pp. 90–95, Springer, Berlin, Germany, 2001. View at Publisher · View at Google Scholar